Žilinská univerzita v Žiline
Spoločnosť pre otvorené informačné technológie
Geokomunita, združenie CSTUG
Česká statistická společnost

OTVORENÝ SOFTVÉR VO VZDELÁVANÍ, VÝSKUME A V IT RIEŠENIACH



Zborník príspevkov medzinárodnej konferencie
OSSConf 2024

2. júla – 4. júla 2024
Žilina, Slovensko

Recenzenti:

Blaško, Rudolf, RNDr., PhD., Borčinová, Zuzana, RNDr., PhD., Kaukič, Michal, Mgr., CSc., Kozubík, Aleš, RNDr., PhD., Kvaššay Miroslav, doc. Ing., PhD., Márton Peter, doc. Ing., PhD., Koháni Michal, doc. Ing., PhD., Peško, Štefan, doc., RNDr., PhD., Rybička Jiří, doc., Ing., Dr., Stříž, Pavel, Ing., Ph.D., Radosław Maciaszczyk, Dr inż., Mirosław Łazoryszczak, Dr inż.

Editori:

Rudolf Blaško, Aleš Kozubík, Pavel Stříž

Všetky práce, uverejnené v zborníku, boli posúdené dvomi nezávislými recenzentmi.

Za jazykovú úroveň zodpovedajú autori príspevkov.

Obsah

Vážení priatelia otvoreného softvéru,

po niekoľkoročnej vynútenej pauze sa vám opäť dostáva do rúk zborník, ktorý je výstupom v poradí už dvanástej samostatnej konferencie, venovanej slobodnému otvorenému softvéru a jeho využitiu vo vzdelávaní, vede a ostatných oblastiach, ktoré si vyžadujú IT riešenia. Aj napriek tomu, že tento ročník konferencie je v poradí dvanásty, po prerušení našej série ako keby sme znovu začínali. Naša konferencia však stále ostáva prvou a zároveň jedinou konferenciou na Slovensku, venovanou otvoreným technológiám.

Tento ročník konferencie prináša jednak tradičné sekcie „LATEX a jeho priatelia“, ktorú sme tento rok rozšírili na „LATEX, R priatelia“ so zdôraznením programovania v prostredí R, ktoré je priateľské LATEX-u, „Open GIS“, „OSS vo vede a vzdelávaní“ a taktiež „Open Hardware“, ktorá sa sľubne rozvíja. Sekcia „LATEXa jeho priatelia“ má dlhodobo svoje pevné miesto na všetkých ročníkoch a najmä zásluhou neúnavnej aktivity Pavla Stříža, ako garanta sekcie, patrí medzi príspevkovo najbohatšie sekcie. Vďaka spolupráci s Českou štatistickou spoločnosťou, ktorú sme nadviazali v rámci medzinárodného projektu Erasmus+ strategické partnerstvo „Innovative Open Source Courses for Computer Science curriculum“, sme ju mohli rozšíriť aj o programovanie v prostredí R. Takisto sekcia „Open GIS“ nechýbala ani na jednom ročníku OSSConf. Tá je doménou nášho priateľa z prostredia `Geokomunity.sk` Mila Ofúkaného. Ani bez jeho úsilia si už nevieme konferenciu ani predstaviť. Treťou z tradičných sekcií je „OSS vo vede a vzdelávaní“, ktorá je asi obsahovo najrôznorodejšia a často v sebe ukrýva niektoré zaujímavé príspevky, ktoré len ťažko dokážeme zaradiť do niektorej inej sekcie. Svoje pevné miesto si našla, aj sekcia „Open Hardware“, ktorá na rozdiel od predchádzajúcich je venovaná otvorenému hardvéru.

Ako spomienku na naše učiteľské semináre sme v programe konferencie tiež pripravili jeden workshop. Bude venovaný znázorňovaniu a interpretácii dát s využitím nástroja marimo notebook. Aj tu sme stavili na osvedčených rečníkov a workshop povedie jeden z „otcov zakladateľov“ našej konferencie Michal Kaukič. Pravidelným účastníkom je zrejmé, že na pozadí všetkého môžu očakávať python.

Ako sa hovorí, pokrok nezastavíš, a tak sa nám stalo, že počas našej prestávky sa stala všadeprítomným fenoménom umelá inteligencia, alebo ak chcete, tak zaužívané označenie AI. Preto sme sa aj my rozhodli otvoriť novú samostatnú sekciu venovanú problematike AI. V tomto duchu chceme pri príležtosti nášho stretnutia rozprúdiť aj diskusiu na tému „Ako ďalej s AI vo vzdelávaní“. Chceli by sme využiť práve rôznorodú skladbu našich účastníkov jednak z rôznych stupňov vzdelávania, ale aj účastníkov z praxe na neformálnu výmenu predstáv a názorov, ako využívať túto technológiu, aby nakoniec neviedla ku pravému opaku, ktorým je prirodzená hlúposť.

Touto cestou by som sa chcel osobitne poďakovať vedeniu Fakulty radenia a informatiky ŽU zastúpenému osobou jej dekana pána prof. Ing. Emila Kršáka, PhD., za dlhoročnú podporu našej konferencie a bezodplatné poskytnutie konferenčných miestností a laboratórií v priestoroch fakulty.

Na záver chcem popriať všetkým účastníkom veľa poučenia a príjemných zážitkov v komunite priaznivcov otvoreného softvéru a otvorených technológií.

Za organizačný výbor OSSConf2024

Aleš Kozubík
predseda

12. ročník
Žilina 2. – 4. 7. 2024
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSS Conf

VIZUALIZACE DAT O ÚMRTNOSTI V ANGLII POMOCÍ SOFTWARU R A INTERPRETACE SROVNÁNÍ MĚR ÚMRTNOSTI U OČKOVANÝCH A NEOČKOVANÝCH PROTI ONEMOCNĚNÍ COVID-19

VENCÁLEK Ondřej (CZ)

VYŽIADANÁ PREDNÁŠKA

Abstrakt: Mohlo by se zdát, že epidemie onemocnění COVID-19, která silně po několik let ovlivňovala naše životy, patří definitivně minulosti. Zatímco se mediální pozornost přesunula k jiným, aktuálnějším problémům, vědecká komunita nadále zkoumá různé aspekty týkající se efektivity jak nefarmakologických opatření typu uzavření škol či sportovišť, tak také léčebných postupů a zejména prevence onemocnění. Hlavní pozornost je přitom zcela pochopitelně věnována efektivitě vakcín, o nichž se hovořilo jako o „jízdence ven z pandemie".

V příspěvku se zamyslíme nad tím, co můžeme říci z veřejně dostupných dat o efektivitě vakcín proti onemocnění COVID-19. Efektivitu přitom můžeme uvažovat proti samotné nákaze, proti těžkému průběhu vyžadujícímu hospitalizaci a proti úmrtí na COVID-19. Zaměříme se přitom právě na efektivitu, s jakou vakcíny brání úmrtí. Prozkoumáme data o úmrtnosti z Anglie, která zveřejnil Britský národní statistický úřad (Office for National Statistics) v srpnu 2023 [1]. Ukážeme několik vizualizací těchto dat, která se objevila na sociálních sítích. Vysvětlíme, v čem je úskalí interpretace těchto grafů.

Ukážeme, jak data vizualizovat pomocí softwaru R [2], a to zejména s využitím balíčku ggplot2! [3], který v současnosti představuje zřejmě nejpopulárnější grafický nástroj softwaru R. K práci s daty využijeme balíček dplyr [4].

V diskusi vysvětlíme rozdíl mezi dvěma základními typy studií, pomocí nichž zkoumáme efekty různých faktorů na lidské zdraví. Jde o experimentální a observační studie. Velká část (ne-li většina) našich poznatků týkajících se efektivity vakcín proti onemocnění COVID-19 je vyvozena ze srovnání nemocnosti či úmrtnosti očkované a neočkované populace. Například výše zmíněná data o úmrtnosti v Anglii jsou observační povahy. To s sebou ovšem nese riziko různých zkreslení.

V současnosti se živě diskutuje o možném zkreslení známém jako Healthy Vaccinee Bias (zkreslení zdravého vakcinovaného), viz např. [5]. Objevuje se tento efekt i v datech o úmrtnosti v Anglii?



doc. Mgr. Ondřej Vencálek, Ph.D. Matematik, statistik, učitel. Pro praktiky teoretik, pro teoretiky praktik. Vystudoval obor pravděpodobnost a matematická statistika na Matematicko-fyzikální fakultě Univerzity Karlovy, kde po magisterském studiu (zakončeném v roce 2007) pokračoval ve studiu doktorském, které ukončil v prosinci 2011 obhájením disertace z oblasti neparametrické statistiky a klasifikace. Zájem o praktické použití statistiky v oblasti medicíny ho vedl k působení na oddělení biostatistiky Státního zdravotního ústavu, v roce 2009 se však vrátil z Prahy na Moravu a začal působit na Univerzitě Palackého v Olomouci.

Těší ho, má-li možnost proslovit popularizační přednášku, jako třeba tu v Planetáriu v Brně (https://www.youtube.com/watch?v=4jE_b3nenQo) nebo sepsat své myšlenky v článečku s lehce filosofickým nádechem (https://www.sisyfos.cz/old/files/Zpravodaj-1-2017.pdf, https://www.statspol.cz/wp-content/uploads/2022/05/IB_2_2022.pdf).

Od roku 2019 je předsedou České statistické společnosti, jejíž činnost se snaží propagovat. Patří mezi popularizátory bayesovských metod, spoluzakládal v současnosti již neexistující uskupení 4BIN. V průběhu epidemie onemocnění Covid-19 se snažil poukazovat na potřebu kvalitního sběru dat a následných nezaujatých analýz.

Literatúra

[1] Deaths by vaccination status, England. Office for National Statistics [online]. [cit. 2024-06-07]. Dostupné z https://www.ons.gov.uk/peoplepopulationandcommunity/birthsdeathsandmarriages/deaths/datasets/deathsbyvaccinationstatusengland.

[2] R Core Team (2023). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria.

[3] Wickham, H.: *ggplot2: Elegant Graphics for Data Analysis*, Springer-Verlag New York, 2016. [cit. 2024-06-07]. Dostupné z https://ggplot2-book.org/.

[4] Wickham, H., François, R., Henry, L., Müller, K., Vaughan, D.: *A Grammar of Data Manipulation*, (2023), R package version 1.1.3.

[5] Fürst, T., Bazalová, A., Fryčák, T., Janošek, J.: *Does the healthy vaccinee bias rule them all?* Association of COVID-19 vaccination status and all-cause mortality from an analysis of data from 2.2 million individual health records. Int J Infect Dis. 2024 May; 142:106976. doi: 10.1016/j.ijid.2024.02.019. Epub 2024 Feb 22. PMID: 38401782.

ANSIBLE: AUTOMATIZÁCIA SPRÁVY SYSTÉMOV A SOFTVÉRU

FEDORIK Slavko (SK)

VYŽIADANÁ PREDNÁŠKA

Abstrakt: Kto spravuje len jeden (svoj) počítač/server, pravdepodobne nikdy neriešil problém automatizácie jeho nastavenia. Dokonca i niekoľko strojov sa dá spravovať manuálne, no tu sa už iste mnohí stretli so situáciou, keď treba tú istú činnosť (príkaz, súbor...) urobiť opakovane na každom. A ako to v takých prípadoch býva, na niektorom príkaz spustiť zabudnete, na inom si nevšimnete, že skončil chybou, a výsledkom je, že máte systémy v stave, ktorý nie je jednoduché ani definovať, ani zistiť.

Na riešenie tohoto problému mnohí používajú spoločné úložisko konfigurácie, alebo multiplexery SSH, ktoré spúšťajú rovnaký príkaz na viacerých strojoch, či iné podobné riešenia, ktoré správu viacerých strojov síce významne uľahčujú, ale často neriešia problém, že stroje sa líšia, a to v rôznych aspektoch, či už majú rôzne OS ale i jeho rôzne verzie, a nezabudnime na ich rôzne úlohy (webový server, DB server, desktop...), ktoré vyžadujú rôzny softvér a zase i jeho rôzne verzie.

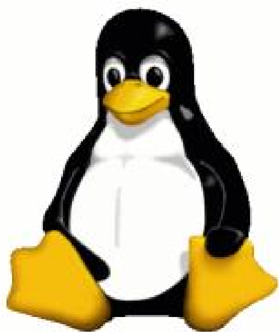
Skutočné riešenie sa nazýva automatizácia konfigurácie/nasadenia. Ja som sa o ňu začal zaujímať po tom, ako som začal používať kontajnery (LXC), ktoré razom zvýraznili rozličnosť ich úloh, ale zároveň vyžadovali značné množstvo spoločných nastavení. Existuje viacero riešení tejto automatizácie, a ja som zvolil Ansible. Neviem, či je lepšie alebo horšie ako ostatné riešenia, zvolil som ho pretože to je slobodný softvér, je dostupný v Debiane ako balík a nevyžaduje agentov. No a aj preto, že používa Python a Jinja2, ktoré sú mi blízke.

Ansible je nástroj (alebo presnejšie sada nástrojov), ktorá poskytuje spôsob nastavenia systému, nasadenia a nastavenia softvéru deklaratívnym spôsobom. Čiže definujete čo má urobiť, na ktorom stroji a za akých podmienok a nemusíte (aspoň teda väčšinou) definovať ako to urobiť. Ako má byť ktorá úloha urobená definuje samotné Ansible, pomocou zabudovaných modulov, ktorých je naozaj nepreberné množstvo, a ak niečo chýba, nie je problém definovať vlastný modul, prípadne nainštalovať ho z externého zdroja (napr. GitHub, Ansible Galaxy, apod.)

Pomocou Ansible možno spravovať stroje kompletne (tzn. všetko okrem inštalácie) alebo i čiastočne (tzn. len niektoré jeho časti/funkcie) a správa viacerých

(i rôznych) strojov prestáva byť nočnou morou otravných opakujúcich sa činností a znovu sa stáva zábavou, ktorá robí radosť… A na jeho použitie nepotrebujete vôbec poznať/vedieť programovanie v Python, no určite využijete znalosť šablón Jinja2.

V príspevku sa pokúsim Ansible predstaviť, popísať jeho základné princípy a možnosti, doplnené o niekoľko ukážok a dúfam, že poslúži ako motivácia na jeho použitie. Samozrejme, pridám aj niekoľko ukážok konfigurácie i výsledku. Zároveň v ňom však ponechám veľa oblastí/funkcií nedotknutých, pretože možností je naozaj veľa, no našťastie, pokojne ich každý môže objavovať postupne, tak ako bude postupne rásť počet úloh, ktoré budete do správy pridávať.



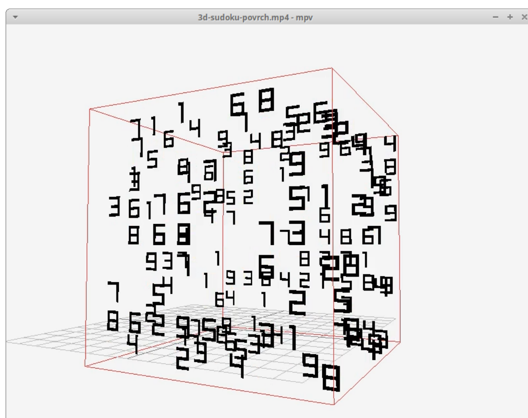
Ing. Slavko Fedorik. Pôvodne vyštudoval vysokofrekvenčnú elektrotechniku a s počítačmi začal pracovať v 80-tych rokoch 20. storočia, najprv ako koníček, no od roku 1997 (dodnes) profesionálne. Viac ako dve desaťročia spravuje Linuxové systémy, desktopy i (verejné a interné) servery. Viac-menej ako koníčku sa venuje programovaniu (v súčasnosti najmä Python) rôznych systémových a sieťových nástrojov a nevyhýba sa ani programovaniu mikrokontrolérov.

Okrem toho dlhoročne prispieva k vývoju slobodného/otvoreného softvéru, najmä v oblasti jeho lokalizácie a autori ho (ne)milujú kvôli schopnosti nachádzať a hlásiť v programoch chyby, ktorých identifikácia vyžaduje dlhodobú a systematickú analýzu. Ako jeden zo zakladajúcich členov stál pri zrode SOIT a hoci zo zdravotných dôvodov musel svoje aktivity výrazne obmedziť, stále propaguje OSS v článkoch (v slovenčine) na svojej webovej stránke (`http://www.slavino.sk`), šíri osvetu o OSS i prácou s mladou generáciou a zasadzuje sa v oblasti počítačovej bezpečnosti a ochrane súkromia (nie len) v počítačových sieťach.

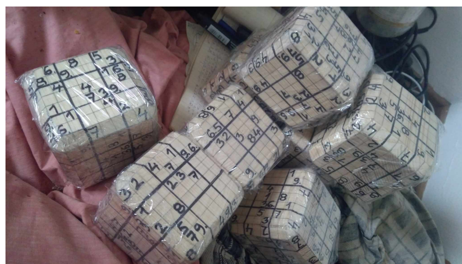
SUDOKU S PŘEKRYVY A JINÉ STŘÍŽOVINY

STŘÍŽ Pavel (CZ)

VYŽIADANÁ PREDNÁŠKA



Abstrakt: Bude mi velkou ctí vám představit mou práci v oblasti rekreační matematiky. Od mého bádání u pozastavení rekurzivní funkce, kde to vše začalo, přes generování a ověřování sudoku s překryvy (kombinované sudoku). Dojde také na řešení sudoku buňku po buňce s komentářem, která hráčská strategie (logická metoda) je použita. Mezi nástroji se objeví programy Sugen (Daniel Beer), Picat (Hakan Kjellerstrand) a Sudoku (Kyle Gough). K automatizaci autor užil Bash, Python, Lua, C a LuaLaTeX. Jádrem povídání bude si představit podmínky kladené na specifická sudoku: olympijské kruhy (PF2024 pro Informační bulletin České statistické společnosti), logo firmy Husqvarna (dárek pro tatíka k narozeninám), měsíční kalendář (vzpomínka na TEXovou legendu doktora Karla Horáka) a 6 sudoku na krychli (jistý experiment, jestli by to vůbec šlo). Autor se snaží najít zobecněnou metodu vypsání podmínek, postupně představí dílčí úvahy a mezikroky s jejich mantinely. Tím se povídání dostane k nejtěžším typům klasického sudoku: 3D sudoku a dvouročnímu kalendáři. Autor představí některé možnosti vizualizace (viz fotky), a nápady, kde a jak bádat nad sudoku dál.



Dárek, který budou moci získat zájemci na konferenci v Žilině.
Byl to jistý výrobní experiment.
Nejspíš však není nad papír.



Ing. Pavel Stříž, Ph.D. (Pája). Programování se Pája věnuje od dětství, ale až na stará kolena zjišťuje, že to nemusí být jen hračka, ale užitečná pomůcka, která hračky dokáže vytvářet. Dokonce dochází k závěru, že programování je něco jako duševní sport: je potřeba se mu zodpovědně věnovat a denně cvičit. Ve svých volných chvílích se zabývá především deskovými (šachy, go) a karetními (bridž, spider) hrami, ale dochází také na hry logické (sudoku). V posledních letech čelí svým letitým nočním můrám: výpočtům kolem Rubikovy kostky a programovacímu jazyku C. Zjistil, že to přináší své ovoce. Rád se věnuje bastlení, opravování elektra a turistice. Ke své práci potřebuje ticho, ale když nepracuje, tak zní vzduchem jemné tóny internetového rádia `http://death.fm/`. Jeho nejoblíbenějším příkazem je: `apt source`. Občas zapomíná. Ach, ano, abychom nezapomněli, občas TEXuje.

12. ročník
Žilina 2. – 4. 7.
2024
Otvorený softvér vo vzdelávaní,
výskume a v IT riešeniach
ossconf.soit.sk
OSS Conf

OSSConf 2024: 13–18

LaTeX A PUBLIKOVANIE NA INTERNETE

RUDOLF BLAŠKO (SK)

Abstrakt. V dnešnej dobe značne pokročili nielen schopnosti počítačov, ale aj zručnosti ich používateľov. Nastal veľký rozmach webových prehliadačov a hlavne aplikácií, ktoré sú schopné v nich fungovať. V tomto smere nezaostávajú ani LaTeX, TeX a s nimi spriaznené aplikácie a súbory. Rôzne animácie dvojrozmerné aj trojrozmerné už bez problémov dokáže LaTeX exportovať nielen do pdf súborov, ale aj na internet, kde si ich môže užívateľ bez problémov pozrieť vo webovom prehliadači.

Kľúčové slová. LaTeX, webový prehliadač, animácia, Asymptote, animate, media9.

Abstract. Nowadays, not only the capabilities of computers, but also the skills of their users have advanced considerably. There has been a great boom in web browsers and especially applications that are able to function in them. In this regard, LaTeX, TeX and their related applications and files are not far behind. Latex can easily publish various animations two and three-dimensional not only to pdf files, but also to the Internet, where the user can view them without any problems in a web browser.

Keywords. LaTeX, web browser, animation, Asymptote, animate, media9.

Úvod

V dnešnej dobe značne pokročili nielen schopnosti počítačov, ale aj zručnosti ich používateľov. Nastal veľký rozmach webových prehliadačov a hlavne aplikácií, ktoré sú schopné „prevádzkovať“. Dnešní mladí ľudia skoro všetko skúmajú a riešia pomocou mobilného telefónu, preto sa neustále zvyšuje význam aplikácií schopných fungovať v mobilných telefónoch. Dôležitým faktorom sú aj technológie schopné preniesť informácie a činnosť týchto užívateľov práve do mobilného telefónu. To sú napríklad interaktívne odkazy a QR kódy. V tomto smere nezaostávajú ani technológie systému LaTeX.

Na tvorbu QR kódov exituje jednoduchý balíček `qrcode`, dvojrozmerné animácie vytvoríme pomocou balíčka `animate`, ktorý je schopný spojiť a následne riadiť sekvencie obrázkov. Tieto obrázky môžeme ovládať alebo sa môžu spúšťať samostatne a nezávislé od užívateľa. Tento proces môže byť ukončený alebo aj neukončený nekonečný. Na vkladanie multimediálnych súborov je vhodný balíček `media9`, pomocou ktorého môžeme do dokumentu vkladať filmy, zvukové

nahrávky ap. Pomocou tohto balíčka môžeme do dokumentu vkladať aj interaktívne 3D obrázky vytvorené pomocou veľmi dobrého programu na kreslenie 2D a 3D grafov funkcií `Asymptote`.

1. Tvorba QR kódov

QR kódy tvoria veľmi jednoduchý a zároveň veľmi účinný Open Source prostriedok na prenos informácií z tlačenej podoby na elektronické média. Postačí iba fotografický prístroj a vhodný softvér. Tejto problematike som sa už venoval na OSSConf v roku 2015 [2], preto iba stručne zopakujem ako jednoducho vytvoriť funkčný QR kód pomocou, ktorého budeme ďalej distribuovať vhodné informácie. Najjednoduchšie je použiť balíček `qrcode`, ktorý načítame do preambuly pomocou príkazov `\usepackage[forget]{qrcode}` alebo `\usepackage[final]{qrcode}`. Rozdiel medzi parametrami `forget` a `final`, je taký, že pri `final` sa vždy generuje nový obrazec, zatiaľčo pri `forget` sa nový obrazec generuje iba pri zmenách údajov. QR obrázok môžeme generovať priamo v hlavnom dokumente, alebo si ho vygenerovať zvlášť a načítať pomocou príkazu `\includegraphics`.

Pri samostatnom generovaní kódu je najvhodnejšie použiť triedu `standalone`, ktorá automaticky oreže formát výstupného súboru podľa veľkosti generovaného obrázka. Ilustruje to nasledujúci príklad, ktorý generuje QR kód s odkazom na domovskú stránku autora.

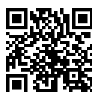
```
\documentclass[10pt]{standalone}
\usepackage[forget]{qrcode}

\begin{document}
  \qrcode[version=0,level=L,height=3cm]{
                 https://frcatel.fri.uniz.sk/users/beerb}
\end{document}
```

Vygenerovaný QR kód načítame do hlavného dokumentu pomocou príkazu `\includegraphics` a pomocou príkazu `\href` ho môžeme v elektronickej forme aktivovať do interaktívnej podoby. V papierovej forme bude kód fungovať po odfotografovaní. Náš súbor s QR kódom sa nazýva `qr1.pdf`.

```
\href{https://frcatel.fri.uniza.sk/users/beerb}{
                    \includegraphics[width=3em]{qr1.pdf}}
```

Výstup predchádzajúceho príkazu bude mať nasledujúci tvar



2. Tvorba 2D animácií

Ak chceme animovať nejaký dvojrozmerný pohyb, musíme ho rozčleniť, analogicky ako na filmovom plátne, na jednotlivé okienka. V praxi to znamená, že pre každú časť pohybu musíme vytvoriť samostatný záber, t. j. samostatný obrázok. Tieto obrázky potom pomocou balíčka `animate` uvedieme do pohybu. Potom môžeme tento pohyb posúvať smerom dopredu alebo dozadu, spomaľovať alebo zrýchľovať, prípadne zastaviť, skočiť priamo na začiatok alebo koniec pohybu ap. Pre tento účel je vhodné do animácie aktivovať riadiace menu.

Postup vytvorenia animácie ukážeme na jednoduchom grafe, ktorý ilustruje prechod hyperkocky medzi jednotlivými dimenziami. Obrázky vytvoríme pomocou balíčka `TiKz`. Uvedený súbor `graf1.pdf` s jednotlivými obrázkami k tejto animácii nájde čitateľ na adrese https://frcatel.fri.uniza.sk/users/beerb/latex/ossconf2024/graf1.pdf



Vygenerovaný zdrojový súbor s obrázkami je síce vo formáte `.pdf`, ale my ho potrebujeme konvertovať do formátu `.svg`, aby sme ho mohli zobrazovať vo webovom prehliadači. Na to existuje v distribúcii `TeXlive` program `dvisvgm`, ale ten konvertuje zo základného výstupného formátu LaTeX-u a to formátu `.dvi`. Tento formát v dnešnej dobe už mládež nepozná, ale starí kozáci a nadšenci áno. Generujeme ho namiesto príkazu `pdflatex` príkazom `latex`.

Obrázky zo súboru `graf1.pdf` animujeme pomocou prostredia `animateinline` v novom súbore, napr. `graf1-web.tex`, následne preložíme do `graf1-web.dvi` a konvertujeme do súboru `graf1-web.svg`. Pred samotnou animáciou potrebujeme vedieť počet obrázkov, ktoré animujeme, aby sme mohli samotný proces automatizovať. V našom prípade uvedený súbor `graf1.pdf` obsahuje 72 obrázkov a súbor `graf1-web.tex` má tvar:

```
\documentclass[dvisvgm]{standalone}
\usepackage{graphicx}
\usepackage{animate}

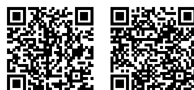
\begin{document}
\begin{animateinline}[autoplay, poster=first, controls, loop,
                            palindrome, buttonsize=14pt]{4}
  \multiframe{72}{ir=1+1}{
              \includegraphics[width=10cm,page=\ir]{graf1.pdf}}
\end{animateinline}
\end{document}
```

Parametre prostredia `animateinline` znamenajú, že sa animácia spusti automaticky od prvého obrázka a potrvá nepretržite (pokiaľ ju nezastavíme) od začiatku do konca a späť, `controls` znamená riadiace menu, jeho veľkosť bude `14pt`. Posledný (povinný) parameter 4 definuje 4 obrázky za sekundu. Hodnota `0.25` by znamenala jeden obrázok za 4 sekundy.

Pre vytvorenie súboru `graf1-web.dvi` použijeme 2-krát príkaz `latex` a následne príkaz `dvisvgm` na konverziu do súboru `graf1-web.svg`.

```
latex graf1-web.tex
latex graf1-web.tex
dvisvgm --font-format=woff --exact --zoom=-1
                                  --page=- graf1-web.dvi
```

Súbor `graf1-web.svg` môžeme priamo spustiť vo webovom prehliadači alebo ho zapúzdriť do `.html` súboru, v ktorom môžeme využiť `javascript` servera `cdn.mathjax.org` na sadzbu online matematiky pomocou LaTeX-u. Do hlavičky `.html` súboru treba pridať nasledovné riadky.



```
<HEAD>
<META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=utf-8">
<TITLE> obrazok </TITLE>
<script type="text/javascript"
  src="https://cdn.mathjax.org/mathjax/latest/
                        MathJax.js?config=TeX-AMS-MML_HTMLorMML">
  MathJax.Hub.Config({tex2jax: {inlineMath: [ [’$’,’$’], ["\\(","\\)"] ],
  displayMath: [ [’$$’,’$$’], [’\[’,’\]’] ] }});
</script>
</HEAD>
```

Ak chceme súbor animovať do `.pdf` súboru, použijeme priamo na danom mieste v texte už spomínaný príkaz

```
\begin{animateinline}[autoplay, poster=last, controls, loop,
                              palindrome, buttonsize=14pt]{4}
  \multiframe{72}{ir=1+1}{
    \includegraphics[width=.5\linewidth,page=\ir]{graf1.pdf}}
\end{animateinline}
```

a dostaneme nasledujúci animovaný obrázok.





3. Tvorba 3D animácií

3D pohyblivé obrázky môžeme vytvoriť viacerými spôsobmi. V tomto príspevku ukážeme ako vytvoriť 3D grafy dvojrozmerných funkcií, ktoré môžeme otáčať, zmenšovať alebo zväčšovať prípadne posúvať. Veľmi vhodným nástrojom pre tvorbu takýchto grafov je Open Source program `Asymptote`, ktorý je priamo integrovaný do prostredia LATEX-u. Bližšie informácie a množstvo príkladov vrátane dvoj aj trojrozmerných animácií aj so zdrojovými súbormi nájde čitateľ na jeho domovskej stránke https://asymptote.sourceforge.io/. Po vložení príkazu

```
\usepackage[inline]{asymptote}
```

do preambuly LATEX-ového zdrojového dokumentu môžeme `Asymptote` scripty písať priamo do textu do prostredia `asy`:

```
\begin{asy}
   zdrojový text pre Asymptote
\end{asy}
```

pričom za príkazom `\begin{asy}` nesmie byť prázdny riadok. Príkaz `\end{asy}` musí byť na samostatnom riadku a musí byť na riadku jediný (ani poznámka pomocou `%` nemôže byť za ním). LATEX dokument sa prekladá na niekoľkokrát [3], preto je vhodné si vytvoriť script na takýto preklad. Napríklad, ak sa náš zdrojový súbor nazýva `subor.tex`, potom preklad by mal prebiehať nasledovne:

```
pdflatex subor.tex
asy subor-*.asy
pdflatex subor.tex
```

Obrázok môžeme samozrejme vytvoriť aj v inom súbore a pomocou balíčka `media9` vložiť do pôvodného súboru, pričom sa zachová jeho pohyblivosť. Takto sme vytvorili súbor `graf2.pdf` s grafom funkcie $z = x \cdot y \cdot (x^2 - y^2)/(x^2 + y^2)$, ktorý čitateľ nájde na adrese https://frcatel.fri.uniza.sk/users/beerb/latex/ossconf2024/graf2.pdf.



Pri preklade do pdf vyzerá preambula `graf2.tex` nasledovne:

```
\documentclass{standalone}
\usepackage[inline]{asymptote}
```

Pri preklade do html je potrebné do preambuly súbora `graf2-web.tex` vložiť:

```
\begin{asydef}
settings.outformat="html";
settings.inlineimage=false;
settings.embed=false;
\end{asydef}
```

a súbor prekladať s parametrom `-v`, t. j.

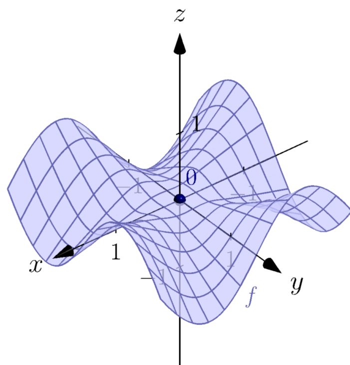
```
pdflatex graf2-web.tex
asy -v graf2-web-*.asy
```

pričom dostaneme súbor `graf2-web-1.html`, ktorý môžeme spustiť priamo v prehliadači.

Vygenerovaný pohyblivý súbor `graf2.pdf` vložíme do textu pomocou príkazu `\includemedia` nasledovne

```
\includemedia[add3Djscript=asylabels.js,
  add3Djscript=3Dspintool.js,activate=onclick,noplaybutton,
  3Dc2c=3 .75 1.75,3Dcoo= 0 0 0,3Droo=150, 3Dortho=.0125]{
    \includegraphics[height=.4\textwidth]{
                                   graf2.pdf}}{graf2-1+0.prc}
```

a dostaneme pohyblivý obrázok (samozrejme nie na papieri), ktorý môžeme ovládať pomocou kurzora myšky.



Poďakovanie. Príspevok vznikol s prispením grantu KEGA 019ŽU-4/2023 „Inovatívne učenie matematiky s podporou Open Source“ podporeného Slovenskou kultúrno-edukačnou grantovou agentúrou.

Literatúra

[1] BRINGHURST, R.: The Elements of Typographic Style, Hartley Marks Publishers 2004, ISBN 0-88179-205-5.

[2] BLAŠKO, R.: *Sadzba QR kódov*, Zborník príspevkov OSSConf 2015, 1.–3. júla 2015, Žilina, ISBN 978-80-970457-7-7.

[3] BLAŠKO, R.: *Asymptote a LaTeX*, Zborník príspevkov OSSConf 2018, 2.–4. júla 2018, Žilina, ISBN 78-80-554-1627-4.

Kontaktná adresa

RNDr. Rudolf Blaško, PhD., Katedra matematických metód a operačnej analýzy, Fakulta riadenia a informatiky, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovensko,
E-mailová adresa: `beerb@frcatel.fri.uniza.sk`

12. ročník Žilina 2. – 4. 7. 2024
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSS Conf

OSSConf 2024: 19–24

INDUSTRIAL SHIELDS. PRIEMYSELNÉ ZARIADENIA ZALOŽENÉ NA ARDUINO A RASPBERRY PI. OSLOBODENIE PRIEMYSLU POMOCOU OPEN SOURCE TECHNOLÓGIE.

RICHARD FABO (SK)

Abstrakt. Článok je voľným pokračovaním (takmer) rovnomennej prednášky z roku 2019. V krátkom úvode uvádza zásadné rozdiely medzi slobodným a neslobodným operačným systémom a softvérom vo všeobecnosti, a tento prístup prenáša i do oblasti programovania PLC automatov. Následne predstavíme niekoľko reprezentantov PLC automatov a HMI zobrazovacích zariadení, ktoré využívajú otvorené a slobodné riadiaci člen používajú procesory Atmega (hoci aj nie zákonite vývojové dosky Arduino), ARM, Xtensa, Broadcom. . . Uvedieme najbežnejšie možnosti (a vývojové prostredia) ich programovania, s dôrazom na bohatý svet slobodného softvéru.

Kľúčové slová. PLC, priemysel, Arduino, ESP32, Raspberry Pi, sloboda.

INDUSTRIAL SHIELDS. INDUSTRIAL DEVICED BASED ON ARDUINO AND RASPBERRY PI. LIBERATING THE INDUSTRY THANKS TO OPEN SOURCE TECHNOLOGY.

Abstract. The article is a free extension of the (almost) namesake lecture from 2019. In a short introduction, it presents the fundamental differences between libre- and non-libre operating systems and software in general, and carries this approach over to the field of PLC programming. We then introduce several representatives of PLCs and HMI display devices that use open and free controllers using Atmega processors (though also not necessarily Arduino boards), ARM, Xtensa, Broadcom. . . We present the most common options (and development environments) for programming them, with an emphasis on the rich world of free (libre) software.

Keywords. PLC, industry, Arduino, ESP32, Raspberry Pi, freedom.

Úvod

Definícií slobody, z rôznych sociálnych, politických alebo technologických hľadísk, je veľa, ale jedna všetko spája – a vraví, že „sloboda je kontrola nad svojím životom“. To vystihuje mnohé, a naznačuje, že čokoľvek, čo túto kontrolu sťažuje alebo znemožňuje, je chápané ako prejav neslobody.

V bežnom živote si to vieme veľmi dobre predstaviť – ostatne najmä sloboda prejavu je stále bolesťou našej civilizácie a ukazuje našu ľudskú vyspelosť.

A čo naša sloboda v IT?

Tá je neustále v ohrození, ale vďaka obrovskej práci vizionárov ako *Richard M. Stallman*, angažovanosti *Free Software Foundation* (a FSFE), projektu *Mozilla*, *Debian*, či mnohých iných (ktorí by si zaslúžili byť menovaní, ale nie je na to priestor) – jednotlivcov či komunít, je táto situácia povzbudivá. Nie, nie je to ideálne, obzvlášť ľudia mimo IT jej nedostatok nevnímajú zásadne, ak vôbec, ale to je práve naša úloha – objasňovať, vysvetľovať, pomáhať.

A v priemysle? Aj tam platí definícia o kontrole? Určite áno.

V oblasti priemyslu, kde je potrebné využívať PC s operačným systémom možno použiť slobodný operačný systém, ako napr. *GNU/Linux.* V oblasti riadenia jednotlivých strojov, procesov, alebo liniek, sa vývojári spoliehajú na *PLC* („Programmable Logic Controller“ je programovateľný logický riadiaci systém, ktorý sa používa na automatizáciu rôznych procesov (nielen) v priemysle a ktoré sa skladá z hardvérovej časti (vstupy, výstupy, procesor) a softvérovej časti (programovacie jazyky)).

Existuje analógia medzi filozofiou slobodného softvéru a otvoreného hardvérového dizajnu (programovateľného pomocou slobodného softvéru). Zjednodušene možno povedať, že kontrola nad (slobodným) softvérom nám umožňuje nielen jeho audit, ale hlavne jeho modifikáciu a distribúciu. Ak toto prenesieme do hardvérového sveta, tak si môžeme predstaviť situáciu, keď napríklad potrebujeme vybaviť jestvujúci systém riadenia o ďalšiu funkcionalitu (napríklad o prepojenie pomocou technológie *LoRa*). Vo svete uzavretého PLC automatu to nie je možné (modifikovať hardvér a softvér pre jednotlivca nie je ani len najmenšou zámienkou na diskusiu). Vo svete otvoreného PLC môžeme hotový LoRa modul pripojiť priamo na piny mikrokontroléra (napr. ESP32) a využiť komunitou vytvorených knižníc na jeho ovládanie. (Toto naštastie v prípade otvorených PLC od Industrial Shields robiť nemusíme, spravia to za nás pri montáži zariadenia.)

Týmto chceme hlavne demonštrovať, že už na úrovni otvoreného hardvérového dizajnu zariadení, sa nám otvárajú obrovské možnosti jeho úpravy, rozšírenia, prispôsobenia. A to nehovoriac o slobodných programovacích nástrojoch. Tým myslíme jednak množstvo, komunitou, ale i výrobcami vytváraných, knižníc, postupov, návodov, ale tiež i sloboda vo výbere programovacieho prostredia (IDE). To je mimoriadne dôležité – ak si uvedomíme, že programátor môže byť navyknutý na konkrétne vývojové prostredie (napr. Vim, Emacs, Geany, VSCode. . .), na jeho ovládanie, klávesové skratky, interakciu so shellom. To všetko sú nezanedbateľné výhody, ktoré zvyšujú efektivitu a komfort práce.

Slobodný softvér a slobodný dizajn hardvéru je predvoj revolúcie v oblasti duševného vlastníctva, najmä v mysliach ľudí na celom svete. Slobodný softvér je dôležitý, pretože oslobodí koncových užívateľov od obmedzení komerčného softvéru, softvérových patentov, DRM a reverzného inžinierstva.

A najmä – slobodný softvér je dôležitý kvôli jeho vzdelávaciemu potenciálu.

Vo vzdelávacom prostredí sa častokrát presadzuje myšlienka, že študenti by sa mali učiť pracovať so zariadeniami, s ktorými sa najčastejšie stretnú v praxi. Tento prístup má svoju logiku. Nie je na škodu, ak si študenti vyskúšajú tieto zariadenia. Avšak to nestačí, pretože škola má i *výchovnú a sociálnu funkciu*, cieľom ktorej je jedinec prospešný pre spoločnosť. Jedinec, ktorý je schopný, nezávislý a kooperujúci. Toto však je v príkrom rozpore s neslobodným, uzavretým softvérom. Tam zdieľanie, kopírovanie, štúdium... je zásadné obmedzené. Ale vedomosti by predsa nemali byť utajené, detaily skryté a ich poznanie zakázané, napríklad licenciami. To všetko obmedzuje rozvoj spoločnosti a „zahmlieva" fakty, ako technológie fungujú. Preto si myslíme, že v edukačnom procese by sa slobodné riešenia mali používať prednostne.

Situácia so slobodnými a súčasne komerčne vyrábanými PLC automatmi

Pred mnohými rokmi sme inštalovali do strojov kusovo a „domácky" vyrábané PLC automaty, založené prevažne na procesoroch PIC. Po roku 2005, keď spoločnosť *Arduino SRL* zásadne zjednodušila používanie mikrokontrolérov, vývojom a distribúciou svojich mikrokontrolérov a dedikovaného vývojového prostredia, začali sa objavovať PLC automaty založené na týchto vývojových doskách. Medzi známe a už etablované značky môžeme zaradiť *Controllino*, *Industruino*, zariadenia od *Adafruit*, v neposlednom rade i priamo od *Arduina*,... a patrí medzi ne i *Industrial Shields*.

Portfólio *Industrial Shields* zahrňuje širokú škálu riešení – od malých (čo sa týka počtu vstupov a výstupov), až po veľké riešenia s podporou WiFi, Bluetooth, LoRa, Dali či GSM.

Stručný prehľad Industrial Shields

Malé PLC, založené na procesoroch *ATmega32u4* sú „malé" počtom vstupov a výstupov (napr. 10 + 10), ale už tieto sú silné z hľadiska komunikačných schopností. RS232, RS485, I2C, SPI – to je štandard, ktorý nie je v tejto kategórii bežný. A navyše i v tejto triede sú dostupné prevedenia s GSM, WiFi či Bluetooth.

Pri potrebe veľkého množstva vstupov a výstupov, vhodné sú PLC štandardne vybavené ethernetom. Prevažne s procesormi *ATmega 2560*, *Xtensa dual-core LX6*. A tieto môžu byť s GSM, WiFi, Bluetooth, LoRa, Dali alebo GSM výbavou. Navyše 32-bitové dvojjadrové procesory integrovanej dosky *ESP32* zjednodušia programovanie, ak musí súčasne bežať viac programových vlákien.

Novinkou sú PLC vybavené procesorom *Arm Cortex-M0 SAMD21*, ktoré zjednodušujú riadenie priemyselných procesov pomocou IoT.



Obr. 1. PLC automaty od Industrial Shields – s rôznymi procesormi

Zaujímavosťou sú PLC automaty, v ktorých je riadiacou súčasťou jednodoskový počítač *Raspberry Pi 4B*. Nakoľko sa jedná o plnohodnotný počítač, so všetkými výhodami, tak nie je ťažké pripojiť k nemu rôzne zariadenia, vyznačujú sa jednoduchou a priamočiarou sieťovou konektivitou, prácou so súbormi, zobrazovaním na ľubovoľnom zariadení s HDMI vstupom, či už s dotykovou vrstvou alebo bez. Tento prístup sprístupňuje priemyselné zariadenia i programátorom, ktorí ich môžu programovať v ľubovoľných programovacích jazykoch (Python, Tcl/Tk, Node-RED...), nehovoriac o obrovskom výpočtovom výkone, ktorý vo svete PLC nemá obdoby.

Je potrebné vizualizovať?

Na to existujú riešenia. Buď samotné PLC založené na Raspberry Pi, s pripojenou obrazovkou, alebo dedikované zariadenie – *Panel PC*. Je to zariadenie, opäť vybavené Raspberry Pi, avšak s obmedzenými počtom vstupov a výstupov. Priemyselné robustné riešenie, s krytím IP65.

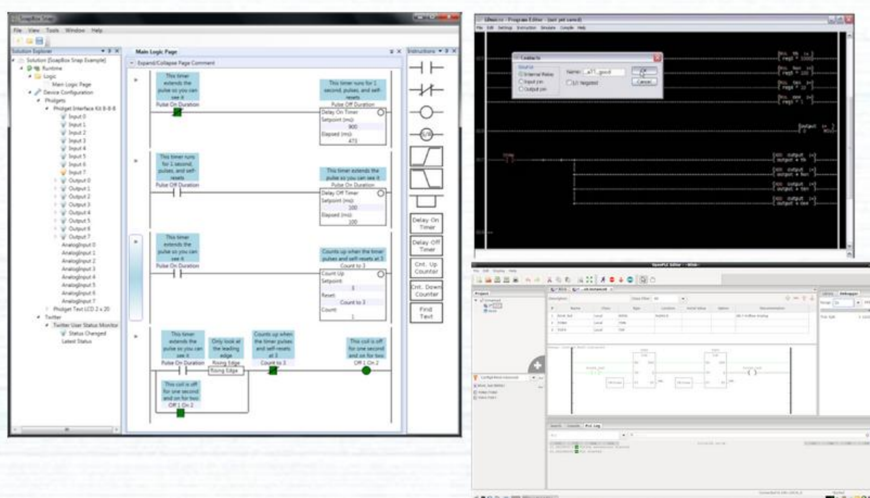


Obr. 2. HMI zobrazovacie zariadenie

Vývojové prostredia

Samozrejme, staré, dobré a osvedčené *Arduino IDE*. Ale nielen to – vďaka programu *arduino-cli* je možné jednoducho a rýchlo implementovať preklad a nahrávanie softvéru do ľubovoľného vývojového prostredia alebo textového editora. Takto programátor ostáva v pohodlí zaužívaného prostredia. Ďalšou možnosťou je programovanie v jazyku *Ladder*, grafických diagramoch.

Spôsob, akým funguje arduino-cli a ako ho integrovať do iného vývojového prostredia je opísaný na: `https://linuxos.sk/blog/richard/series/arduino-a-kamarati-bez-ide/`.



Obr. 3. Programovanie pomocou Ladder diagramov

Záver

Keďže v PLC sú zariadenia zo slobodného sveta otvoreného hardvéru, ako je Arduino či Raspberry Pi, tak informácie, schémy, riešenia sú na dosah vďaka silnej komunite. Komunite, ktorá v slobodnom svete nielen vytvára a spravuje produkty (softvér, hardvér), ale prevádza ich audit, hľadanie chýb a ich nápravu. Toto je možné len v slobodnom svete. A nikdy to nebude možné v proprietárnom svete.

Cena je menej dôležitá – ale je veľmi priaznivá. Je to dôsledok priaznivej licencie, otvoreného a slobodného softvéru a návrhu hardvéru. Eric S. Raymond opísal dynamiku vývoja slobodných a otvorených riešení vo svojej populárnej knihe „The Cathedral and the Bazaar", v ktorej zhodnotil, že pokrok vo vývoji technológií (s otvoreným zdrojovým kódom) je trikrát až štyrikrát rýchlejší

ako pokrok vo vývoji komerčných technológií, vďaka rýchlejšej iterácii, spolupráci a inovácii (čo vedie k rýchlejším vývojovým cyklom a rýchlejšiemu prijatiu nových technológií).

Ukazuje sa, že ak by sme nemali vydobytú slobodu v počítačových vedách, neexistovali by dostupné, výkonné a efektívne zariadenia, aké dnes bežne používame (smartfóny sú zreteľným príkladom).

Najdôležitejšia je sloboda a na nej skutočne záleží. Mať kontrolu nad svojim výrobkom, zariadením, softvérom. Aby sa nestalo, že niekto vymyslí „štandard" alebo protokol, ktorého implementácia bude stáť veľa peňazí. Nemôže sa stať, že budete musieť upgradovať vývojové prostredie, s vysokými nákladmi, aby ste boli schopní spravovať staré a navrhovať nové zariadenia. Zmena operačného systému na vyššiu verziu nie je dostatočným argumentom pre takýto krok.

Nepáči sa vám niečo na vývojovom prostredí, ale nemôžete to zmeniť? Nepáči sa vám niečo na hardvérovom zariadení, ale nedokáže to zmeniť? Núti vás prevádzka či programovanie zariadenia používať neslobodné operačné systémy?

Poslaním tohto článku bolo len naznačiť pár skutočností – cesta k etickému priemyslu je omnoho dlhšia.

Industrial Shields, ale aj iné slobodné riešenia, sú dobrou voľbou. Hovoríme o dospelých a vyzretých produktoch, testovaných v ťažkých priemyselných prostrediach, dodržiavajúce príslušné nariadenia a normy.

To je poslanie komunít, to je stará známa veta: „Sharing is caring".

Kontaktná adresa

Ing. Richard Fabo, Triton Famme s.r.o., Levočská 862/28, 058 01 Poprad,
E-mailová adresa: triton@famme.sk, https://triton.famme.sk

12. ročník Žilina 2. – 4. 7. 2024
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
osscconf.soit.sk
OSS Conf
OSSConf 2024: 25–34

PRIEMYSELNÉ ZARIADENIA ZALOŽENÉ NA ARDUINE, RASPBERRY PI, ESP32... GNU EMACS AKO IDE

RICHARD FABO (SK)

Abstrakt. Článok pojednáva o využití interpertra programovacieho jazyka (e)Lisp so zabudovaným textovým editorom, s názvom GNU Emacs. V krátkom opise sa vyzdvihnú jeho hlavné a najdôležitejšie vlastnosti, oblasti použitia a jeho jedinečnosť vo svete softvéru, z hľadiska funkčnosti i filozofie.

Spomenutím hlavných nedostatkov originálneho Arduino IDE sa dostaneme k spôsobu, ako nahradiť Arduino IDE ľubovoľným textovým editorom (prípade vývojovým prostredím), vo všetkých základných oblastiach (od editovania kódu, cez jeho nahrávanie do vývojových dosiek (nielen Arduino), až po nahradenie sériovej konzoly). Bližšie sa opíše, ako funguje proces inštalácie vývojových dosiek a knižníc, ako s nimi narába arduino-cli, a ako generovať FQBN (plne kvalifikované názvy dosiek) pre účely kompilácie a nahrávania.

Kľúčové slová. PLC, priemysel, Arduino, ESP32, Raspberry Pi, IDE.

INDUSTRIAL DEVICES BASED ON ARDUINO, RASPBERRY PI, ESP32... GNU EMACS AS AN IDE.

Abstract. This article discusses the use of the (e)Lisp programming language interpert with a built-in text editor, called GNU Emacs. A brief description will highlight its main and most important features, its areas of application and its uniqueness in the software world, both in terms of functionality and philosophy.

By mentioning the main shortcomings of the original Arduino IDE, we will get to the way to replace the Arduino IDE with any text editor (or integrated development environment), in all basic areas (from editing code, to uploading it to development boards (not only Arduino), to replacing the serial console). It will describe how the process of installing development boards and libraries works, how arduino-cli handles them, and how to generate FQBNs (fully qualified board names) for compilation and upload purposes.

Keywords. PLC, industry, Arduino, ESP32, Raspberry Pi, IDE.

Úvod

Spoločnosť Arduino SRL zohrala veľkú úlohu pri popularizovaní programovania mikrokontrolérov. Ich platforma `Arduino` ponúka jednoduchý a priateľský spôsob pre začiatočníkov aj skúsených programátorov na vytváranie a experimentovanie s elektronickými projektami. Vďaka ich Open Source prístupu sa programovanie mikrokontrolérov stalo prístupnejším a atraktívnejším pre ľudí po celom svete.

Vydané Arduino IDE umožňovalo (a umožňuje) zjednodušené programovanie vďaka veľkému množstvu knižníc a príkladov, a tiež preklad a nahrávanie kódu do mikrokontroléra bez potreby znalosti tohto procesu. Už v prvých verziách mal integrovaný sériový monitor, zvýrazňoval syntax, umožňoval presun na chyby počas kompilácie...

Proces prekladu bol zásadne zjednodušený, pretože nebolo potrebné vytvárať vlastné Makefile súbory na kompiláciu, linkovanie, nahrávanie, s potrebou špecifikovania veľkého množstva parametrov.

Samotné vývojové prostredie bolo v porovnaní so zavedenými „dospelými" vývojovými prostrediami (Vim, Emacs, Visual Studio, Eclipse, Sublime Text, Atom ai.) okliešténé a chýbali mu vlastnosti, ktoré zjednodušujú a zrýchľujú proces programovania, napríklad:

- dopĺňanie kódu (z predchádzajúcich výskytov alebo z slovníka);
- „folding" (zbaľovanie) častí kódu;
- presun na definície, funkcie;
- integrovaná dokumentácia;
- práca v konzole ai.

Niektoré chýbajúce funkcie a vlastnosti a zlepšený výkon prinieslo až Arduino IDE 2.0 v roku 2021.

1. Arduino-cli

Arduino-cli je nástroj pre príkazový riadok, ktorý umožňuje kompilovať, nahrávať a spravovať knižnice (a dosky) Arduino (a iné) bez potreby prostredia Arduino IDE. Zjednodušene môžeme povedať, že „obaľuje" (zastrešuje) programy potrebné k prekladu a nahrávaniu kódu, volá ich s požadovanými parametrami, číta ich stavy. Vďaka rozhraniu príkazového riadku je ho možné využiť v rôznych vývojových prostrediach. My sa zameriame na GNU Emacs.

1.1. Štruktúra adresárov pre arduino-cli, nástroje

Arduino-cli vyžaduje, aby určité súbory (napr. definície vývojových dosiek, binárne spustiteľné súbory pre kompiláciu a pod.) boli umiestené v presne definovanej adresárovej štruktúre. Tento článok bude opisovať skutočnosti v operačnom systéme GNU/Linux.

Po inštalácii arduino-cli (viď https://arduino.github.io/arduino-cli/) a zavolaní príkazu arduino-cli config init dôjde k vygenerovaniu súboru arduino-cli.yaml. V našom prípade vyzerá jeho časť nasledovne (máme ho presunutý v skrytom adresári ~/.arduino15)

```
board_manager:
  additional_urls:
```

```
    https://arduino.esp8266.com/stable/
            package_esp8266com_index.json
    http://apps.industrialshields.com/main/arduino/boards/
           package_industrialshields_index.json

directories:
  data: /home/richard/.arduino15
  downloads: /home/richard/.arduino15/staging
  user: <cesta k~našim programom pre vývojové dosky>
```

Za zmienku stoja URL adresy v sekcii board_manager, ktoré určujú, odkiaľ sa budú sťahovať príslušné json súbory pre iné ako originálne (Arduino) vývojové dosky a zariadenia. V našom prípade máme pridané dosky pre ESP8266 a PLC zariadenia Industrial Shields (https://famme.sk/plc.html).

V sekcii directories sú nami definové cesty ku knižniciam a nástrojom pre všetky inštalované vývojové dosky a tiež k našim programom. Viac informácii k štruktúre adresárov, ktoré by presiahli rozsah tohto článku, môže láskavý čitateľ nájsť na https://linuxos.sk/blog/richard/detail/programovanie-arduino-a-derivatov-bez-arduino/.

Základné príkazy arduino-cli pre prácu s doskami

- zoznam inštalovaných dosiek arduino-cli board listall,
- aktualizácia zoznamu výrobcov a dosiek arduino-cli core update-index,
- inštalácia novej dosky arduino-cli core install <výrobca>:<doska>,
- vyhľadávanie v zoznamoch dosiek arduino-cli core search <reťazec>.

1.2. Čo je FQBN a ako ho vytvoriť?

FQBN je skratka z Fully Qualified Board Name („plne kvalifikovaný názov dosky“). Ide o jedinečný identifikátor konkrétnej dosky, ktorý zvyčajne obsahuje typ dosky, model mikrokontroléra, taktovaciu frekvenciu a ďalšie dôležité informácie. Je to jedným z nevyhnutných parametrov zadávaných pre arduino-cli, ktorý sa skladá z nasledovného:

```
<výrobca>:<architektúra>:<označenie dosky>[=<ďalšie možnosti,
                                      napr. typ procesora, frekvencia>]
```

Niekoľko príkladov FQBN:

- Arduino Leonardo arduino:avr:leonardo,
- Arduino Uno WiFi arduino:avr:unowifi,
- Industrial Shields Ardbox Analog HF+
  industrialshields:avr:ardbox:cpu=ardboxanaloghfplus232,
- Industrial Shields M-Duino 58+
  industrialshields:avr:mduino:cpu=mduino58plus.

Ako zistiť tieto FQBN?

Čitateľa v prípade záujmu o podrobnejšie informácie, ako sa tieto FQBN tvoria, ako ich manuálne zistiť zo štruktúry súborov inštalovaných dosiek, len odkážeme na odkaz `https://linuxos.sk/blog/richard/detail/programovanie-arduino-a-derivatov-bez-arduino-2/`. Nám nateraz stačí, že FQBN si vieme vygenerovať pomocou skriptu v jazyku `Tcl`, dostupného na adrese `https://gitlab.com/tcl-tk-public/arduino-fqbn-generator`.

Skript nás prevedie viacerými krokmi:

- v 1. kroku zistí a vypíše zoznam dosiek, ktoré máme nainštalované;
- po zvolení dosky (defacto výrobca) zistí, aké máme jej verzie;
- po zvolení verzie zistí, či je potrebné ešte upresnenie typu cpu – ak áno, tak je možné si tento zvoliť v následnom kroku;
- nakoniec vypíše vygenerovaný FQBN a tento uloží do súboru `fqbn.txt`, na ktorý sa odkazuje `Makefile`.

Skript spustíme príkazom:

```
tclsh <cesta ku skriptu>/arduino-fqbn-generator.tcl.
```

Pre spustenie skriptu potrebujeme mať nainštalovaný interpreter jazyka `Tcl`. S najväčšou pravdepodobnosťou, hraničiacou s istotou, bude `Tcl` v repozitároch bežnej GNU/Linuxovej distribúcie, napr. v distribúciach založených na projekte `Debian` ho nainštalujeme príkazom `sudo apt install tcl`.

1.3. `Makefile` pre kompiláciu a nahrávanie programov

Ako bolo spomínané, na tento účel je možné v GNU Emacs použiť rozšírenie `arduino-cli-mode`. Osobne však preferujem vytvorenie `Makefile` súboru.

`Makefile` je súbor obsahujúci pravidlá a akcie určené na automatizáciu procesu kompilácie a spustenia programov v rámci vývojového prostredia. Tento súbor definuje závislosti medzi zdrojovými kódmi a objektovými súbormi, ako aj príkazy potrebné na kompiláciu a linkovanie programu. Náš `Makefile` vyzerá nasledovne (krátené, celý súbor na `https://gitlab.com/tcl-tk-public/arduino-fqbn-generator/-/tree/main/makefile_for_arduino-cli`):

```
PROGRAM ?= $(wildcard ./*.ino)

ifeq ($(wildcard /dev/ttyACM*),)
    ifeq ($(wildcard /dev/ttyUSB*),)
        PORT ?=
    else
        PORT ?= $(firstword $(wildcard /dev/ttyUSB*))
    endif
else
    PORT ?= $(firstword $(wildcard /dev/ttyACM*))
```

```
1 *eshell* 2 arduino-bez-IDE-4.org 3 arduino-fbqn-generator.tcl 4 *eshell*<3>
5 *eshell*<2> +
-----------------------------------------

Zoznam inštalovaných dosiek:
0 → arduino
1 → esp8266
2 → industrialshields
3 → rp2040

➲ Zadaj číslo dosky:
2

Zoznam variantov dosiek industrialshields:
0 → Ardbox DALI family                   industrialshields:avr:ardboxdali
1 → Ardbox GPRS family                   industrialshields:avr:ardboxgprs
2 → Ardbox LoRa family                   industrialshields:avr:ardboxlora
3 → Ardbox WiFi/BT family                industrialshields:avr:ardboxwifi
4 → Ardbox family                        industrialshields:avr:ardbox
5 → M-Duino DALI family                  industrialshields:avr:mduinodali
6 → M-Duino GPRS family                  industrialshields:avr:mduinogprs
7 → M-Duino LoRa family                  industrialshields:avr:mduinolora
8 → M-Duino WiFi/BT + GPRS family        industrialshields:avr:mduinowifigprs
9 → M-Duino WiFi/BT family               industrialshields:avr:mduinowifi
10 → M-Duino family                       industrialshields:avr:mduino
11 → Spartan family                       industrialshields:avr:spartan

➲ Zadaj číslo varianty dosky:


U:**-  *eshell*<3>    Bot L32    ( )
```

Obr. 1. Generátor `FQBN`

```
endif

FQBN ?= $(shell cat fqbn.txt)
# \dotskontrola, či existuje súbor fqbn.txt

ARDUINO-CLI-BIN ?= /home/richard/.arduino15/arduino-cli
# \dotskontrola, či existuje arduino-cli

kompiler-verify ?= $(ARDUINO-CLI-BIN) compile -b
kompiler-verbose ?= $(ARDUINO-CLI-BIN) compile -v -b
```

```
kompiler-upload ?= $(ARDUINO-CLI-BIN) upload -v -b

all:
 $(ARDUINO-CLI-BIN) compile -b $(FQBN) $(PROGRAM)

upload:
ifeq ($(PORT),)
 $(error Chyba: zariadenie nepripojené!)
else
 @echo "Zariadenie pripojené na: $(PORT)"; \
 $(ARDUINO-CLI-BIN) upload -v -b $(FQBN) -p $(PORT) $(PROGRAM);
endif
```

Vysvetlenie kódu:

- `PROGRAM ?= $(wildcard ./*.ino)`
  Vytvoríme si premennú `PROGRAM`, do ktorej si načítame názov súbor s príponou `ino` (môže byť len jeden taký súbor v adresári s Makefile, inak ho treba explicitne definovať).
- `ifeq ($(wildcard /dev/ttyACM*),)\dots`
  `ttyACM`, resp. `ttyUSB` sú systémové súborové symbolické odkazy, ktoré sa používajú na komunikáciu s USB zariadeniami pripojenými k počítaču cez sériový port.
- `FQBN ?= $(shell cat fqbn.txt)`
  Načítanie plne kvalifikovaného názvu dosky, vytvoreného skriptom v jazyku `Tcl`.
- `ARDUINO-CLI-BIN ?= home/richard.arduino15/arduino-cli`
  Je cesta k binárke `arduino-cli`.
- `kompiler-verify ?= $(ARDUINO-CLI-BIN) compile -b`
  `kompiler-verbose ?= $(ARDUINO-CLI-BIN) compile -v -b`
  `kompiler-upload ?= $(ARDUINO-CLI-BIN) upload -v -b`
  Sú pomocné premenné, aby sme nemuseli reťazce, ktoré reprezentujú, písať celé v neskorších receptoch. Tie sa totiž konštruujú tak, aby boli ľahko upraviteľné práve zmenami premenných.
- `all, upload`
  Sú konkrétne „recepty“, t. j. časti kódu, ktoré sa vykonajú, ak zavoláme z príkazového riadku príkaz `make`. „Recept“ `all` sa vykoná, ak je príkaz `make` bez parametrov, `verbose`, ak použijeme príkaz `make verbose` atď.

Ak to zhrnieme, postup pri zostavovaní programu pomocou `arduino-cli` s využitím `Makefile` je nasledovný:

- inštalácia príslušnej vývojovej dosky pomocou
  `arduino-cli core install <výrobca>:<doska>`;

- spustenie skriptu na vygenerovanie FQBN pomocou `tclsh <cesta ku skriptu>/arduino-fqbn-generator.tcl`;
- zostavenie programu pomocou `make` (prípadne `make verbose`, pre viac informácii počas kompilácie).

Následne nahratie do pripojeného zariadenia sa vykonáva pomocou príkazu `make upload`.

2. GNU Emacs

Niektoré definície zhrňujú:

> „GNU Emacs je textový editor so zameraním na prácu s textom a programovaním, ktorý bol vytvorený Richardom Stallmanom v roku 1984 ako súčasť projektu GNU. Je jedným z najstarších a najznámejších textových editorov vo svete s voľne dostupným zdrojovým kódom.“

Richard Stallman pracoval na MIT (Massachusetts Institute of Technology), kde sa tešil programovací jazyk `LISP` veľkej popularite. Je preto pochopiteľné, že napísal malé jadro editora v jazyku `C`, ktorého hlavný účel spočíval v interpretácii dialektu jazyka `lisp` špeciálne určeného na editovanie textu, ktorý dostal názov `Elisp` (uvádzaný tiež ELISP, elisp, (e)lisp). Vznikla tak nová (dnes sa nám zdá samozrejme prirodzená), zaujímavá paradigma vo vývoji softvéru – malé rýchle jadro + skriptovací jazyk.

V Emacse sa akákoľvek interakcia vykonáva pomocou príkazov. A tieto príkazy sú jednoduché Elisp-ové funkcie.

Hlavnou dátovou štruktúrou Emacsu sú `buffery`. Na pozadí sú spôsobom výmeny textových prúdov. V hľadiska ovládania je výhodou, že editačné a vizuálne príkazy a funkcie sú vždy rovnaké – či sa jedná o písanie kódu, alebo prácu napríklad v emulátore terminálu.

Dôležitou súčasťou sú tzv. režimy, nazývané `módy`. Je to súbor nastavení a dodatočných funkcií, ktoré menia správanie buffera a teda aj spôsob interakcie s užívateľom. Definujú také veci, ako napr. odsadenie textu, zvýrazňovanie syntaxe, priradenia klávesových skratiek.

Pochopiteľne, tento článok nie je a nemôže byť návodom na používanie Emacs-u v plnom jeho rozsahu, s využitím všetkých funkcií a možností, ktoré nám poskytuje interpreter jazyka Elisp.

2.1. arduino-mode, arduino-cli-mode

`arduino-mode` (`https://melpa.org/#/arduino-mode`) je rozšírením `c-mode` GNU Emacs-u o potrebné konštanty, kľúčové slová a úpravu odsadzovania.

2.2. Editačné funkcie, chýbajúce v Arduino IDE

Vzhľadom na rozsah článku nie je možné opísať všetky funkcie, ktoré robia Emacs unikátnym. Preto si len zdôraznime niektoré, ktoré nám editáciu kódu zásadne zlepšia. Za zmienku stoja:

Skoky na definície premenných a referencie funkcií (s náhľadom alebo bez), s využitím viacerých spôsobov: pomocou vygenerovaného indexu zdrojového kódu programom `ctags` resp. `etags`; pomocou interaktívneho zužovania textu počas zadávania vyhľadávaného reťazca (príkaz `M-x helm-swoop`) v aktuálnom bufferi alebo v rámci celého projektu (príkaz `M-x noccur`).

Prepojenie s databázou dokumentácie príkazov. S využitím projektu `Dash`, ktorý umožňuje získavanie offline dokumentácie, a jej následné zobrazenie vo vstav internetovom prehliadači; tiež je možné túto databázu využiť ako základ pre doplňovanie zadávaných príkazov pomocou slovníka (`https://linuxos.sk/blog/richard/detail/gnu-emacs-vzdy-je-co-vylepsit-2-dash/`).

Emulátor terminálu (vytvorený v Elisp-e). Jeho hlavnou výhodou, oproti zaužívaným emulátorom terminálu, je možnosť využívania všetkých editačných príkazov Emacs-u (voľný pohyb kurzora, doplňovanie, vyhľadávanie, priame spúšťanie funkcií či programov napísaných v Elisp-e, vstavaný nástroj na vzdialený prístup cez ssh, ftp. . .).

„Zbaľovanie“ výrazov, funkcií, vizuálne potlačenie časti kódu. Známa funkcia aj z iných vývojových prostredí zásadne zlepšuje orientáciu v kóde. Vizuálne potlačenie časti kódu (funkcie `M-x narrow-*`), na ktorom práve nepracujeme, zjednodušuje orientáciu a zlepšuje sústredenie.

Automatické doplňovanie (z predchádzajúcich výskytov, zo slovníka). Pomocou niekoľkých, navzájom sa doplňujúcich, funkcií je možné text doplňovať postupným stláčaním klávesovej skratky, alebo výberom z „drop-down“ ponuky.

Strom súborov a funkcií . Je grafickým zobrazením štruktúry celého projektu, s jednoduchým presunom po súboroch projektu, funkciách a definíciách. Nakoľko sa (stále) jedná o buffer Emacs-u, tak máme k dispozícii všetky jeho editačné funkcie.

„Snippety“ –vytvorenenie časti kódu po zadaní kľúčoveho slova. „Snippety“ (viď nNami vytvorené snippety, spolu so slovníkom pre automatické doplňovanie na `https://gitlab.com/arduino91/arduino-yasnippets-for-emacs`) sú fragmenty kódu alebo textu, ktoré je možné vytvoriť a uložiť do databázy a potom ich kedykoľvek vložiť do textu zadaním krátkeho kľúčového slova. Takto možno automatizovane vytvoriť i komplikované štruktúry, bez potreby ich pamätania.

Virtuálne stránky. Slúžia na rozdelenie kódu po ľubovoľne dlhých častiach, po ktorých je možné sa efektívne presúvať (napr. príkazom `M-x helm-pages`).

Viacnásobné kurzory sú šikovnou funkciu, ktorá umožňuje mať viacero kurzorov v texte naraz. Táto funkcia umožňuje editovať alebo zmeniť viacero častí textu naraz pomocou jedného pohybu kurzora. Tento nástroj je užitočný pre editovanie textových súborov, kde je potrebné vykonať rovnakú zmenu na viacerých miestach.

Záložky a registre. Záložky, vizuálne, v rámci jedného súboru, alebo ich databáza (obmedzená len na pripojené súborové systémy) slúžia na rýchle presuny na uložené pozície. Pritom táto pozícia nemusí byť len v rámci jedného súboru, alebo súborov v rámci jedného projektu, ale môže sa jednať i o presun do adresárov, pozície v PDF súbore, riadok v terminále, poznámku v tele e-mailu a pod. Registre sú viacnásobné schránky, uložené pod ľubovoľnými názvami, ktoré môžu uschovávať text, napríklad časť kódu. Vkladanie obsahu registrov môže sprevádzať i náhľad na ich obsah.

Dynamické expandovanie výberu je mimoriadne návyková funkcia. V podstate sa jedná o rozširovanie výberu textu, pričom po aktivácii klávesovej skratky sa zvolí slovo, následne celý výraz, blok, funkcia.

Správa verzií. Jedná sa o systém, ktorý umožňuje užívateľovi sledovať zmeny v texte a ukladať históriu verzií dokumentu, umožňuje zaznamenávať zmeny, porovnávať verzie a obnovovať predchádzajúce verzie dokumentu. Emacs podporuje rôzne systémy, ako sú Git, Mercurial, CVS, Bazaar atď.

Rýchly presun a kopírovanie v rámci viditeľnej časti. Jedná sa o presun kurzora na viditeľný text pomocou rozhodovacieho stromu založeného na znakoch. Jeho rozšírením je kopírovanie oblastí na pozíciu kurzora. Snahou je minimalizovanie počtu stlačení klávesov pri týchto častých operáciách (`https://linuxos.sk/blog/richard/detail/avy-utok-velociraptora/`).

Elektrické a farebné zátvorky Elektrické zátvorky sú funkcia, ktorá automaticky vkladá uzatváraciu zátvorku (rôznych typov) pri písaní otváracej zátvorky. Pomáha to udržiavať zátvorky správne uzavreté a minimalizovať prípadné chyby. Farebné zobrazenie rôznych zátvoriek pomáha vizuálne odlíšiť jednotlivé zanorenie blokov kódu, nakoľko sa zátvorky zobrazujú v zjednotených farbách, a teda je ľahké identifikovať, ktoré zátvorky sa k sebe vzťahujú.

Integrovaný „serial monitor“ . Vstavaný sériový monitor, pri práci s ktorým opäť využívame všetky editačné možnosti Emacs-u.

Preklad a nahrávanie Samotný Emacs obsahuje príkaz `M-x compile`, ktorý zavolá `make`, v prípade potreby s doplnenými parametrami.

Na nahrávanie kódu nám postačí malá funkcia:

```
(defun arduino-make-upload (&optional arg)
  "Upload to device using Makefile"
  (interactive)
  (shell-command (concat "make upload")))
```

Kontaktná adresa

Ing. Richard Fabo, Triton Famme s.r.o., Levočská 862/28, 058 01 Poprad,
E-mailová adresa: triton@famme.sk, https://triton.famme.sk

12. ročník
Žilina 2. – 4. 7.
2024
Otvorený softvér vo vzdelávaní,
výskume a v IT riešeniach
ossconf.soit.sk
OSS Conf

OSSConf 2024: 35–42

GNU EMACS + ORG MÓD = EASY LATEX.

RICHARD FABO (SK)

Abstrakt. Článok pojednáva o využití rozšírenia Org-mode pre GNU Emacs na zjednodušenú tvorbu dokumentov v LATEX-u. Využitie značkovacieho jazyka s jednoduchým ovládaním, bez potreby zadávania príkazov, interaktívnou tvorbou textových tabuliek, mnohými exportnými funkciami (ai.) je výhodné pre písanie dokumentov bez rozptyľovania sa jeho formátovaním. Je vhodný pre tých, ktorí dokumenty v LATEX-u píšu zriedkavo, bez potreby pamätania si mnohých jeho príkazov. Súčasne sa tento text jednoduchšie edituje, a možno na neho aplikovať všetky prednosti, ktoré ponúka editor GNU Emacs, od kontroly pravopisu až po správu verzií.

Kľúčové slová. Emacs, LATEX, org mode, značkovací jazyk.

GNU EMACS + ORG MODE = EASY LATEX.

Abstract. This article discusses the use of the Org-mode extension for GNU Emacs to simplify document creation in LATEX. The use of a markup language with simple controls, without need to enter commands, interactive creation of text tables, and many export functions (among others) is advantageous for writing documents without the distraction of formatting them. It is suitable for those who rarely write documents in LATEX, without the need to remember many of its commands. At the same time, this text is easier to edit, and all the advantages offered by the GNU Emacs editor can be applied to it, from spell checking to version control.

Keywords. Emacs, LATEX, org mode, markup language.

Úvod

Písanie v jednoduchom textovom dokumente, bez formátovania, pomáha sústrediť sa na samotný text a nerozptyľovať sa vizuálnymi prvkami. Absencia formátovania umožňuje sústrediť sa výlučne na obsah a myšlienky, ktoré chceme vyjadriť, bez starania sa o estetiku. To vedie k efektívnejšiemu procesu písania. Isteže, formátovacie funkcie sú pri dlhšom texte nevyhnutné, avšak tvorca textu nemusí byť zahlcovaný vzhľadom tlačového výstupu. Historický program `WordPerfect`, až do verzie 5.1 pracoval iba v textovom režime počítača a prípadné formátovanie rozlišoval vizuálne zmenou farieb textu, prípadne pozadia. Tento prístup sa neskôr vytratil (pričom možno diskutovať, či sa jednalo viac o cnosť programu, alebo obmedzenie samotného programu a vtedajšieho hardvéru), do popredia sa dostal režim WYSIWYG (`What You See Is What You`

Get), teda spôsob zobrazenia obsahu na obrazovke, kde to, čo je viditeľné počas tvorby obsahu, je (v prevažnej miere) to, ako bude obsah vyzerať aj po jeho zverejnení alebo tlači. Značkovacie jazyky, ako AsciiDoc, Markdown alebo Org naproti tomu preferujú prístup WYSIWYM, teda „vidíš to, čo máš na mysli“.

1. Čo je to Org-mode?

Org-mode, označovaný aj org-mode, ORG, org je rozšírením GNU Emacs-u,[1] ktoré slúži na organizáciu a spravovanie textových informácií. Je určené na písanie poznámok, úloh, plánovanie projektov a iné organizačné účely. Org-mode umožňuje vytvárať hierarchické zoznamy, sledovať časové harmonogramy a má ďalšie funkcie, ktoré umožňujú efektívnejšiu prácu s textom. Org-mode tiež poskytuje možnosť exportu dokumentov do rôznych formátov (HTML, PDF, LaTeX, ODT…). Dajú sa v ňom efektívne tvoriť tabuľky, dá sa „previazať“ s inými programami. Dá sa povedať, že to nie je len značkovací jazyk, ale efektívne prostredie na ľubovoľnú prácu s informáciami v textovej podobe.

2. Syntax dokumentu v Org-mode (pre LaTeX dokumenty)

V Org-mode existuje špeciálna sekcia (časť dokumentu), ktorá sa používa na skrytie dodatočných informácií alebo obsahu, nazývaná drawer. Táto sekcia je vizuálne skrytá z pohľadu užívateľa, ale môže byť podľa potreby a požiadavky zobrazená. Vloženie tejto sekcie v dokumente, ktorý sa má exportovať do LaTeX-u sa vykonáva zadaním kľúčového slova :HEADER: (resp. príkazu org-insert-drawer) a je ukončená reťazcom :END:. Môže vyzerať nasledovne (krátené):

```
:HEADER:
#+LATEX_CLASS: article
#+LATEX_CLASS_OPTIONS: [a4paper]
#+LATEX_HEADER: \usepackage{lmodern}
#+LATEX_HEADER: \usepackage{subcaption} \usepackage{listings}
#+TITLE: *Demo \LaTeX{} dokument vytváraný v /org mode/*
#+DATE: 2024-05-16 Št
#+OPTIONS: toc:t num:t title:t
:END:
```

Z príkladu je zrejmé, že:

- #+LATEX_CLASS: article, #+LATEX_CLASS_OPTIONS: [a4paper], #+LATEX_HEADER: \usepackage{lmodern}, atď., sú príkazy, ktoré sa vložia do preambuly .tex dokumentu, pričom v jednej tejto direktíve, ktorých počet nie je obmedzený, možno použiť viac LaTeX-ových príkazov oddelených medzerou,
- #+OPTIONS: toc:t num:t title:t a #+OPTIONS: d:(not "DR2") sú dodatočné príkazy pre generovanie .tex dokumentu.

[1]Bohatý textový editor, rozšíriteľný pomocou skriptov v jazyku Elisp.

A teda `#+` udávajú interné príkazy a parametre dokumentu v org-mode.

Na mieste je otázka:
„Tak je potrebná znalosť LATEX-u pri práci v org-mode?“

Stručná odpoveď: „Nepochybne.“.
O trochu dlhšia odpoveď: „To záleží na tom, že… “ …doplnená o skúsenosti autora článku „aký typ dokumentov, s akou opakovateľnosťou sa vytvára“.

Ak teda hovoríme o tvorbe dokumentov s využívaním rovnakej šablóny, vzhľadu, určenia, tak potom je počet zadávaných príkazov LATEX-u skutočne minimálny. Na prípadnú korekciu výstupu stačí poznanie niekoľkých základných príkazov (typicky `\newpage` a príkazy na tvorbu matematických výrazov).

Ak to zhrnieme: na tvorbu LATEX dokumentu nám postačuje hlavička súboru (`:HEADER:`) a samotný text.

Jednou z výhod org-mode je aj to, že môžeme ľubovoľne kombinovať formátovacie značky a príkazy org-mode s príkazmi LATEX-u.

3. Ako zadávať a formátovať text?

Zadávanie príkazov definujúcich štruktúru dokumentu (podsekcie, zoznamy…) sa vykonáva zadaním počtu znakov `*` podľa úrovne sekcie.

```
* Nové počítače v ČSSR
** Rok 1968: Na VŠ sa inštalujú prvé samočinné počítače

Počítač /MSP 2/ bol prvým sériovo vyrábaným samočinným počítačom
v Československu.
Na akademické pracoviská a do podnikov
+a domácností+ sa montoval v rokoch *1967 - 1968*.
Vedeckí
pracovníci sa vyjadrovali na jeho adresu pochvalne.
Išlo o
_tranzistorový_ univerzálny počítač, ktorý bol dekadický, sériový,
alfanumerický a pracoval na systéme diernej pásky.
```

Zadanie formátovania (tučný, šikmý text, horný index…) sa vykonáva uzavretím slova (vety, odstavca) do znakov (ako to poznáme napríklad z `Markdown`-u, hoci `org-mode` používa iné znaky), napríklad:

```
*Toto bude tučný text.*
/A toto bude tlačené kurzívou./
```

`Org` nám aj samotnú zmenu formátovania zobrazí graficky, pričom môže zobrazenie uzatvárajúcich znakov (`*`, `/`, `_`…) potlačiť.

Ak si pozrieme vytvorený `.tex` súbor (príkaz `org-latex-export-as-latex`), tak zistíme, že nami zadané znaky sa vyexportovali ako v tab. 1. Analogicky sa prekladajú i zoznamy (nečíslované i číslované) rôznych úrovní.

```
* Nové počítače v ČSSR

** Rok 1968: Na VŠ sa inštalujú prvé samočinné počítače

Počítač MSP 2 bol prvým sériovo vyrábaným samočinným počítačom v Československu. Na
akademické pracoviská a do podnikov a domácností sa montoval v rokoch 1967 - 1968.
Vedeckí pracovníci sa vyjadrovali na jeho adresu pochvalne. Išlo o tranzistorový
univerzálny počítač, ktorý bol dekadický, sériový, alfanumerický a pracoval na
systéme diernej pásky.
```

Obr. 1. Ukážka formátovania s potlačenými uzatvárajúcimi znakmi.

Tabuľka 1. Preklad znakov org-mode na príkazy LaTeX-u

Znak `org-mode`	LaTeX-ový príkaz
`* <text>`	`\section{<text>}`
`** <text>`	`\subsection{<text>}`
`- <položka>`	`\item <položka>`
`/<text>/`	`\emph{<text>}`
`*<text>*`	`\textbf{<text>}`
`_<text>_`	`\uline{<text>}`
`+<text>+`	`\sout{<text>}`
`[fn:<číslo>]`	`\footnote[<číslo>]{<text>}`

4. Tabuľky

Tabuľky a ich tvorba sú veľmi silným argumentom, prečo používať `org-mode`. Ako vidíme na priloženom obrázku 2, tvorba tabuľky sa vykonáva zadaním znaku `|` a medzery, opakovaného podľa počtu stĺpcov. Nové bunky sa vytvárajú klávesou `TAB`, `\hline` pomocou `- - -`. Pritom úprava šírky a výšky buniek je automatická, podľa obsahu zadaného v bunkách tabuľky, taktiež môže byť šírka bunky explicitne daná (počtom znakov).

```
#+CAPTION: Numerické operácie v tabuľke
#+NAME: tab2
#+ATTR_LATEX: :placement [!htbp]
#+ATTR_LATEX: :align | c | c | r | r | r |
| p.č. | typ | počet jadier | cena za jadro | spolu |
|------+-----+--------------+---------------+-------|
|    1 | LO  |            4 |            50 |   200 |
|    2 | MI  |            8 |            40 |   320 |
|    3 | HI  |           16 |            30 |   480 |
#+TBLFM: $5=$4*$3
```

Obr. 2. Tvorba tabuľky

Ďalšie vlastnosti tabuľky (názov, referencia, zarovnanie...) docielime zadaním príkazov `#+CAPTION`, `#+NAME`, `#+ATTR_LATEX`.

Autor sa pokúša naznačiť, že tvorba tabuľky z ASCII znakov je úplne intuitívna, jednoduchá a automatická.

Príjemnou vlastnosťou je možnosť zadávania vzorcov a týmto využiť `org-mode` ako dobre použiteľný tabuľkový procesor (najmä ak uvážime, že vzorce možno zadávať i ako funkcie jazyka Elisp, so všetkými výhodami (i nevýhodami)).

5. Obrázky

Vloženie obrázku sa vykonáva príkazom `org-insert-link`. Ako názov príkazu napovedá, dajú sa takto vložiť i iné formy odkazov (súbor, internetový odkaz...). Predvolená klávesová skratka `C-o C-l` je mnemotechnická, a vždy v Emacs-e platí, že klávesová skratka sa dá priradiť k (úplne) každému príkazu, funkcii, makru, atď.

```
#+CAPTION: Samočinný počítač MSP 2A
#+NAME:obr1
#+ATTR_LATEX: :width 15cm :options angle=0
file:obr1.jpg
```

Obr. 3. Vloženie obrázku s dodatočnými parametrami

```
#+CAPTION: Samočinný počítač MSP 2A
#+NAME:obr1
#+ATTR_LATEX: :width 15cm :options angle=0
```

Obr. 4. Vloženie obrázku a jeho (nastaviteľný) náhľad

Ako vidíme na obrázkoch 3 a 4, pomocou direktívy `#+CAPTION` zadáme jeho opis, `#+NAME` referenčný názov a pomocou `#+ATTR_LATEX` ďalšie parametre ako veľkosť, umiestnenie, otočenie... Navyše príkazom `org-toggle-inline-images` môžeme zapnúť či vypnúť zobrazenie obrázkov priamo v bufferi `org-mode`.

6. Org-mode + Gnuplot

`GNUPLOT` asi netreba predstavovať, stručne sa dá povedať, že je to slobodný program na tvorbu grafov a vizualizáciu dát, vybavený vlastným skriptovacím jazykom, kompatibilný s `POSIX štandardmi`.

V `Emacs`-e je dlhodobo prítomné rozšírenie s názvom `gnuplot-mode`, ktoré slúži na editáciu skriptov spomínaného programu. Funkcie tohto rozšírenia môžu byť volané z `org-mode`, a vstupnými dátami môže byť tabuľka vytvorená priamo v `org-mode`. Stručný príklad:

Majme tabuľku s náhodnými číslami:

```
#+CAPTION: Náhodné čísla
#+NAME: tab3
#+PLOT: title:"Tab. 3" ind:1 deps(2:3) type:2d
```

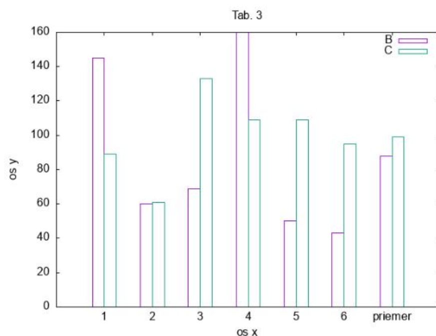
```
        with:histograms set:"yrange [0:]"
#+PLOT: set:"xlabel 'os x'"  set:"ylabel 'os y'"
        set:"linewidth 20"
#+PLOT: labels:("p.č." "B" "C")  file:"graf1.png"

|    p.č. |   B |   C |
|---------+-----+-----|
|       1 | 145 |  89 |
|       2 |  60 |  61 |
|       3 |  69 | 133 |
|       4 | 160 | 109 |
|       5 |  50 | 109 |
|       6 |  43 |  95 |
|---------+-----+-----|
| priemer |  88 |  99 |
#+TBLFM: @>$2=vmean(@2..@-1);%.0f :: @>$3=vmean(@2..@-1);%.0f
```

Pomocou niekoľkých parametrov `#+PLOT` môžeme upresniť vzhľad výsledného grafu, definovať rady údajov a mnohé iné vlastnosti výsledného grafu. V našom prípade sa po zavolaní funkcie `org-plot/gnuplot` (ak je kurzor v tabuľke alebo jej hlavičke) vytvorí súbor `graf1.png`, ktorý môže byť použitý v inom mieste súboru. Výstup z tejto tabuľky je na obrázku 5.

p.č.	B	C
1	145	89
2	60	61
3	69	133
4	160	109
5	50	109
6	43	95
priemer	88	99

Tabuľka 3: Náhodné čísla



Obr. 3: Vygenerovaný graf z predchádajúcej tabuľky **(3)**

Obr. 5. Tabuľka, z ktorej sme spravili graf a ten následne použili v texte

7. Náhľad na PDF dokument v okne Emacsu

Spomínaný WordPerfect (do verzie 5.1), StarWriter a hoci aj v našich končinách populárny Text602, mali možnosť grafického náhľadu na výsledný dokument. Org-mode disponuje niekoľkými exportnými funkciami a jednou z nich je aj export do PDF súboru, pomocou funkcie org-latex-export-to-pdf. Postupné vyvolanie menu a jeho položky sa vykonáva klávesovou skratkou C-e l o. Dde-facto sa jedná o sekvenciu stlačenia klávesov, čo môže byť nepohodlné a navyše sa PDF dokument otvára v externom prehliadači, predvolene MuPDF.

Vygenerovaný PDF súbor môžeme v druhom okne prehliadať jednoduchou funkciou:

```
(defun make-pdf-from-org-via-latex (&optional arg)
  "Make PDF and display it via LaTeX{}"
  (interactive)
  (save-buffer)
  (latex-popup-message "\nProsím čakaj.
                         \n\nPDF sa generuje")
  (org-latex-export-to-pdf)
  (switch-to-buffer  (concat
                     (file-name-base (buffer-name)) ".pdf"))
  (revert-buffer t t)
  (previous-buffer)
  (latex-popup-close))

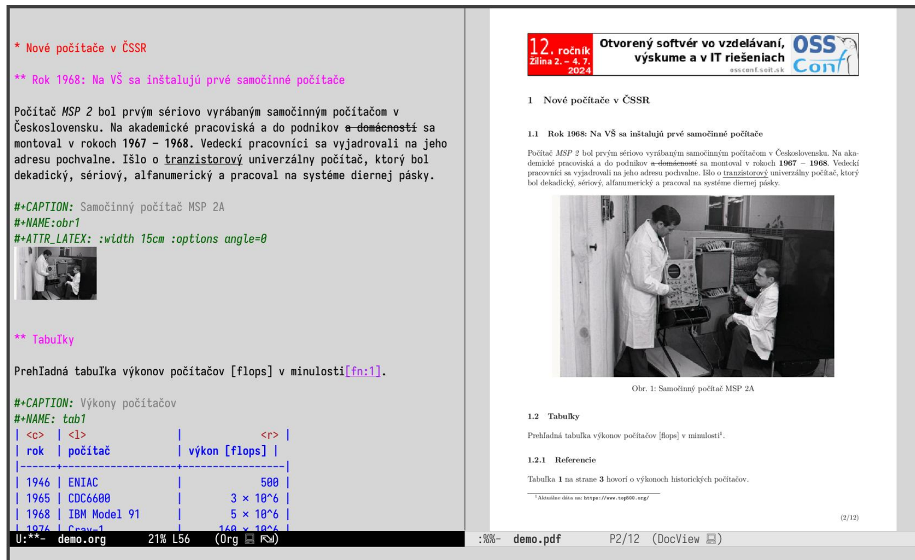
(define-key org-mode-map (kbd "<f9>")
 'make-pdf-from-org-via-latex)
```

Navyše, ak máme aktívny zvolený región alebo zobrazenú len časť dokumentu (funkcie narrow-to-*), generuje sa nám len preambula a viditeľná/označená časť dokumentu, čo môže byť výhodné pri veľkých súboroch.

8. Záver

Opísať viac možností org-mode by niekoľkonásobne presiahlo rozsah tohto článku a hádam aj únosnú mieru množstva prijatých informácií.

Jedným z cieľom článku bolo i naznačiť skutočnosť, že kombináciu Org + LaTeX možno použiť aj ako každodennú náhradu textového procesora. Zdanlivá nevýhoda - linkovanie externých obrázkov – je pri dobrej organizácii naopak výhodou z hľadiska veľkosti zdrojových súborov. O prenositeľnosti, čitateľnosti, editácii a v neposlednom rade o i estetike vytváraných dokumentov nie je potrebné diskutovať.



Obr. 6. Náhľad na vyexportovaný PDF súbor

Kontaktná adresa

Ing. Richard Fabo, Triton Famme s.r.o., Levočská 862/28, 058 01 Poprad, Slovensko, *E-mailová adresa*: `triton@famme.sk`, `https://triton.famme.sk`

12. ročník Žilina 2. – 4. 7. 2024
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSS Conf
OSSConf 2024: 43–50

AKO MI POMOHOL BALÍČEK tcolorbox

ALEŠ KOZUBÍK (SK)

Abstrakt. Pri príprave knižných publikácií sa často stáva, že je potrebné niektoré pasáže zvýrazniť, opticky oddeliť od ostatného textu. Vhodným nástrojom sú rozličné orámovania, ktoré dokážu zvýrazniť takéto objekty v dokumente. Práve pri príprave učebnice autor objavil balíček `tcolorbox`, ktorý ho zaujal svojou jednoduchosťou a tým aj rýchlosťou dosiahnutia požadovaného efektu. V tomto článku prinášame jemný úvod do práce s balíčkom od vytvorenia najjednoduchšieho rámca až po náročnejšie dekorované objekty.

Kľúčové slová. LaTeX, rámčeky, balíček `tcolorbox`.

HOW PACKAGE TCOLORBOX HELPED ME

Abstract. When preparing book publications, it frequently happens we need to highlight or optically separate some passages from the rest of the text. Suitable tools are various frames that can highlight such objects in the document. It was during the preparation of the textbook that the author discovered the `tcolorbox` package, which impressed him with its simplicity and at the same time the speed of achieving the desired effect. The present article brings a gentle introduction to the work with the package, from creating the simplest frame to more complex decorated objects.

Keywords. LaTeX, frameboxes, `tcolorbox` package.

Úvod

Pri sadzbe rozsiahlejších dokumentov ako sú knihy, monografie či učebnice, ale aj bežné učebné materiály, sa stretneme s rôznymi požiadavkami na úpravu vzhľadu stránky. Učebné texty si neraz vyžadujú možnosť zvýraznenia určitých častí dokumentu ich optickým oddelením od ostatného textu. To môžu byť napríklad definície, vety, zdrojové kódy a podobne.

Samotný systém LaTeX ponúka možnosti orámovania textu pomocou príkazu `\framebox` resp. `\fbox`, ako sa uvádza napríklad v [5] alebo v [3]. Takto realizované rámčeky sú ale trochu sparťanské, čo obstojí v striktne vedeckých textoch, avšak v učebniciach vyžadujeme trochu pestrejšie vyhotovenie rámov a boxov. Všeobecne možno konštatovať, že čím je text publikácie menej exaktný, tým vyššie sú požiadavky na pestrosť jeho grafickej úpravy. To isté platí aj pre učebné texty pre nižšie vzdelávacie stupne, ako aj pre prezentácie alebo postery.

Ak teda chceme dokument opticky spestriť, máme dve možnosti. Buď pôjdeme cestou dôvery iba k vlastným makrám a pokúsime sa sami s využitím grafických možností balíčka `TikZ`, alebo nebudeme znovu objavovať koleso a poobzeráme

sa po dostupných riešeniach. Pracujúc v časovej tiesni sa zväčša rozhodneme pre druhú možnosť, ktorú v tomto prípade prináša balíček tcolorbox. Jeho autorom je Thomas F. Sturm a podrobný manuál [7] nájdeme online v archíve CTAN.

Pochopiteľne, pre prácu je potrebné zvládnutie práce v typografickom systéme TEX-LATEX aspoň v rozsahu krátkych úvodných článkov [1], [2] a učebníc [5] alebo [6]. Podrobnejší výklad pravidiel prostredia TEX potom nájdeme v [3]. Pretože balíček dobre spolupracuje s balíčkom TikZ, je užitočné oboznámiť sa aj s prácou s týmto balíčkom aspoň v rozsahu článku [4] z predchádzajúcich ročníkov našich konferencií.

1. Začíname s balíčkom tcolorbox

Balíček tcolorbox môžeme jednoducho popísať tak, že poskytuje prostredie pre vytváranie farebných, orámovaných boxov so samostatným riadkom určeným pre hlavičky boxov. Je súčasťou typickej inštalácie systému TEX-LATEX, takže si nevyžaduje žiadne doplňujúce inštalácie. Pre jeho použitie ho do dokumentu vložíme štandardne pomocou príkazu \usepackage, teda:

```
\usepackage{tcolorbox}
```

Pri načítaní balíčka tcolorbox sa automaticky načítajú aj ďalšie balíčky pgf, verbatim, etoolbox a environ. Okrem toho môže byť balíček načítaný s voliteľnými argumentmi, ktoré určujú použité knižnice, napríklad:

```
\usepackage[listings]{tcolorbox}
```

čím načítame knižnicu pre úhľadnejšie zobrazenie zdrojových kódov. Alternatívne je možné knižnice načítať v preambule dokumentu príkazom \tcbuselibrary, teda napríklad

```
\tcbuselibrary{listings}
```

Najjednoduchší box vygenerujeme pomocou prostredia tcolorbox, takže kód by vyzeral takto:

```
\begin{tcolorbox}
  Toto je jednoduchý rámček.
\end{tcolorbox}
```

Dostaneme tak výsledok, ktorý sa zatiaľ veľmi neodlišuje od bežného frameboxu.

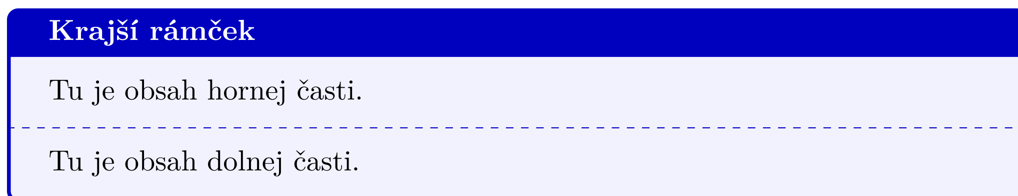
Toto je jednoduchý rámček.

To môžeme zmeniť tým, že zadefinujeme nejaké pomenovanie rámčeka, zmeníme jeho farebnosť a prípadne oddelíme hornú a dolnú časť obsahu. Farby a nadpis zadáme ako voliteľné parametre prostredia tcolorbox, ako ilustruje nasledujúci zdrojový kód:

```
\begin{tcolorbox}[colback=blue!5!white,colframe=blue!75!black,
    title=\textbf{Krajší rámček}]
```

```
    Tu je obsah hornej časti.
  \tcblower
    Tu je obsah dolnej časti.
  \end{tcolorbox}
```

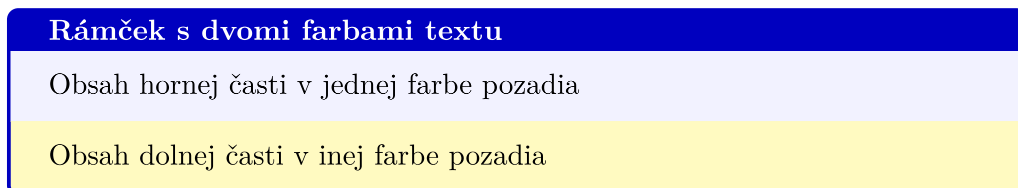
Výsledkom je potom takýto farebný rámik:



Všimnime si, že s farbami pracujeme rovnako ako pri všetkých ostatných farebných objektov, a to vrátane ich zmiešavania. Upozornime aj na príkaz `\tcblower`, ktorý oddeľuje hornú a dolnú časť obsahu rámčeka. Túto deliacu čiaru je možné tiež zobraziť pomocou príkazu `\tcbline`, ktorý je súčasťou knižnice `skin`. Na rozdiel od `\tcblower` tento príkaz nepredstavuje rozdelenie obsahu boxu na dve funkčné jednotky, ale iba zobrazí čiaru a je ho možné použiť aj niekoľkokrát. Rozdiel sa prejaví najmä v situácii, kedy chceme zmeniť vzhľad hornej a dolnej časti rámca. Tak napríklad pomocou kódu

```
\begin{tcolorbox}[skin=bicolor,colback=blue!5!white,
    colframe=blue!75!black,title=\textbf{Rámček s dvomi farbami textu},
    colbacklower=yellow!30]
  Obsah hornej časti v jednej farbe pozadia
\tcblower
  Obsah dolnej časti v inej farbe pozadia
\end{tcolorbox}
```

vytvoríme nasledujúci rámček s odlišným pozadím v jeho hornej a dolnej časti.



To by pri oddelení pomocou `\tcbline` nebolo možné. Tiež si všimnime spoluprácu s voľbou `skin=bicolor`, ktorá je pre zmenu farebnosti kľúčová.

2. Vybrané voliteľné parametre prostredia tcolorbox

Kompletný prehľad voliteľných `pgf` parametrov je možné nájsť v manuáli [7]. V tomto odstavci uvedieme len vybrané, najpoužívanejšie parametre. Už sme videli, že voľbou `title=<text>` zadávame nadpis rámca. Môže sa však stať, že

nie vždy budeme s výsledkom spokojní, najmä ak zobrazíme viacero rámčekov vedľa seba:

prvý	**druhý**	**Tretí**	**Štvrtý**
Nejaký text.	Nejaký text.	Nejaký text.	Nejaký text.

Ako vidíme, výška nadpisov a tým aj polí rámcov nie sú rovnaké, čo je nevzhľadné a rušivé. Tento nedostatok odstránime použitím voliteľného parametra `adjusted title`.

Potom dostaneme takýto výsledok:

prvý	**druhý**	**Tretí**	**Štvrtý**
Nejaký text.	Nejaký text.	Nejaký text.	Nejaký text.

Na tomto mieste môžeme zmieniť aj možnosť nastavenia jednotného vzhľadu rámcov príkazom `\tcbset{voľby}`. Tak sa vyhneme opakovanému zadávaniu voliteľných parametrov prostredia.

V predchádzajúcej ukážke sme použili nastavenie vzhľadu voľbami:

```
\tcbset{colback=white,arc=0mm,width=(\linewidth-12pt)/4,
  equal height group=AT,
  before=\hfill,after=\hfill,
  fonttitle=\bfseries,
  colframe=red!75!black}
```

V druhej verzii rámcov sme potom nastavenú farebnosť orámovania „prebili" priamym zadaním parametra `colframe` prostredia `tcolorbox`.

Občas sa stáva, že nadpis rámu je dlhý a presahuje dĺžku jedného riadku. Voliteľným parametrom `squeezed title` zariadime kompresiu tohto nadpisu tak, aby sa zmestil do jedného riadku. Jeho hviezdičkovaná verzia potom kombinuje túto kompresiu spolu s aktiváciou parametra `adjusted title`. Ak teda nastavíme parametre prostredia takto

```
\begin{tcolorbox}[squeezed title*=\n,colframe=blue!75!black]
```

dostaneme ako výsledok:

Nadpis	**Dlhý nadpis**	Ešte dlhší nadpis	Štvrtý veľmi ale veľmi dlhý nadpis
Nejaký text.	Nejaký text.	Nejaký text.	Nejaký text.

Dôležitým prvkom je potom oddeľovanie hornej a dolnej časti rámčeka, čo regulujeme logickou hodnotou parametra `lower separated=false|true`. Zaujímavý efekt potom poskytuje kombinácia rámca s parametrom `beamer`, čo ilustruje nasledujúca ukážka:

Oddelená dolná časť

Toto je horná časť.

Toto je dolná časť.

Bez oddelenia dolnej časti

Toto je horná časť.

Toto je dolná časť.

Príslušné parametre prostredia `tcolorbox` sme potom nastavili takto:

```
\begin{tcolorbox}[beamer,title=Oddelená dolná časť,
  width=(\linewidth-12pt)/2]
```

resp. v pravom rámci:

```
\begin{tcolorbox}[beamer,title=Bez oddelenia dolnej časti,
  lower separated=false,width=(\linewidth-12pt)/2]
```

Ako užitočný nástroj, ktorý poskytuje balíček `tcolorbox` môžeme uviesť prostredie `\tcblistings`. To umožňuje v rámiku ilustrovať zdrojový kód a hneď vedľa neho výsledok jeho použitia. Ilustrujeme si to na ukážke zobrazenia smajlíka pomocou TikZ-u. Výhodou je, že narozdiel od kombinácie prostredia `verbatim` s kódom a následnej ukážky, tu zapisujeme zdrojový kód len raz. Prostredie deklarujeme takto (v preambule je potrebná načítať knižnicu `listings`):

```
\begin{tcblisting}{width=\linewidth-5pt,tikz lower,listing side text,
  fonttitle=\bfseries,bicolor,colback=blue!20,colbacklower=white,
  colframe=black,righthand width=3cm,title=TikZ smajlík}
```

Ako výsledok potom dostaneme nasledujúci rámček.

TikZ smajlík

```
\path[fill=yellow,draw=yellow!75!red]
(0,0) circle (1cm);
\fill[red] (45:5mm) circle (1mm);
\fill[red] (135:5mm) circle (1mm);
\draw[line width=1mm,red]
(215:5mm) arc (215:325:5mm);
```

Samozrejme, pre správnu činnosť je potrebné načítať balíček `TikZ`. So základmi kreslenia v TikZ-e je možné sa oboznámiť napríklad v [4].

3. Dekoratívne prvky rámčekov

Už bola zmienená dobrá spolupráca balíčkov `tcolorbox` a `tikz`. To je možné využiť na vylepšenie rámčekov rôznymi dekoratívnymi prvkami. To sa uplatní najmä pri príprave výučbových materiálov pre mladšie ročníky alebo pri príprave rôznych propagačných prezentácií.

Ukážkou dekorácií môže byť napríklad vytvorenie rámčeka so žabími očami, zdanlivo pozorujúcimi obsah rámčeka spolu s čitateľom.

Žabí box

Žabka zdanlivo pozoruje obsah rámčeka.

Príslušný zdrojový kód potom vyzerá takto:

```
\tcbset{zabka/.style={enhanced,
  colback=green!10,
  colframe=green!65!black, enlarge top by=5.5mm,
  overlay={\foreach \x in {11cm,12.5cm} {
\begin{scope}[shift={([xshift=\x]frame.north west)}]
  \path[draw=green!65!black,fill=green!10,line width=1mm]
  (0,0) arc (0:180:5mm);
  \path[fill=black] (-0.5,-00) arc (0:180:1mm);
\end{scope}}}}}
\noindent
\begin{tcolorbox}[zabka,title=Žabí box, width=\linewidth]
  Žabka zdanlivo pozoruje obsah rámčeka.
\end{tcolorbox}
```

V prvej časti kódu, pomocou príkazu `\tcbset` upravíme vzhľad boxu. Parameter `overlay` potom charakterizuje vonkajšiu dekoráciu, ktorú vytvoríme pomocou kresliaceho balíčka `tikz`.

Inou možnosťou je dekorovanie rámika stužkou, napríklad pre vyznačenie nejakej úlohy, za ktorú študent získa nejaké prémiové ocenenie bodmi alebo známkou.

Opäť, najskôr pomocou príkazu `\tcbset` definujeme štýl rámčeka a pomocou nástrojov balíčka `tikz` nakreslíme stužku v pravom hornom rohu. Definícia štýlu potom môže vyzerať nasledovne:

```
\tcbset{stuzka/.style={enhanced,
  colback=red!5!white,
  colframe=red!75!black,fonttitle=\bfseries,
  overlay={\path[fill=blue!75!white,
  draw=blue,double=white!85!blue,
  preaction={opacity=0.6,fill=blue!80},
  line width=0.1mm,
  double distance=0.2mm,
  pattern=fivepointed stars,
  pattern color=yellow]
  ([xshift=-0.2mm,yshift=-1.02cm]frame.north east)
  -- ++(-1,1) -- ++(-0.5,0) -- ++(1.5,-1.5) -- cycle;}}}
```

A tu je očakávaný výsledok:

Prémiová úloha

Text zadania.

Úlohy, ktoré treba splniť.

Zaujímavým dekoratívnym prvkom rámčeka môže byť umiestnenie vodoznaku do jeho pozadia. Ten môže byť zadaný buď v podobe textu, pri použití voliteľného parametra:

```
watermark tex = <text>,
```

alebo ako obrázok pri použití.

```
watermark graphics = <meno súboru>
```

Ilustrovať si to môžeme napríklad na zobrazení diskografie vybranej kapely. Názvy štúdiových albumov (pre úsporu miesta neuvádzame kompletný) tvoria obsah boxu a logo kapely je umiestnené ako vodoznak do pozadia. Rámček potom dopadne takto:

Diskografia

1974 Kiss
1974 Hotter Than Hell
⋮
2012 Monster



Zdroj: https://www.discogs.com/artist/153073-Kiss

Príslušný zdrojový kód:

```
\tcbset{colback=red!5!white,colframe=red!75!black,fonttitle=\bfseries}
\noindent
\begin{tcolorbox}[enhanced,title=Diskografia,width=\linewidth
  watermark graphics=kiss_f.png,watermark opacity=0.15]
  1974 Kiss\\
  1974 Hotter Than Hell\\
  $\vdots$\\
  2012 Monster
\tcblower
  Zdroj: \url{https://www.discogs.com/artist/153073-Kiss}
\end{tcolorbox}
```

4. Namiesto záveru

V tomto článku sme si predstavili základné prvky balíčka `tcolorbox`, tak že umožňujú, pri znalosti ostatných nástrojov systému TEX/LATEX, pohodlne a pomerne rýchlo vytvoriť v dokumente viac či menej pôsobivé orámované objekty.

Autor sám objavil balíček pri písaní učebnice z oblasti investícií a pod časovým tlakom odovzdania rukopisu ocenil práve rýchlosť preniknutia do základov práce.

Nie je samozrejme jednoduché na niekoľkých stránkach poskytnúť vyčerpávajúci prehľad nástrojov a podrobný návod práce s balíčkom, ktorého manuál [7] má rozsah väčší než 500 strán. Niektoré ďalšie námety a pokročilejšie techniky si preto ponecháme na ilustráciu pri samotnej prezentácii v rámci prednášky. Spomeňme aspoň niektoré ďalšie možnosti dekorovania, vytvárania rámčekov ako plávajúcich objektov, možnosti rotácie rámov.

Možnosti balíčka `tcolorbox` je tiež možné s výhodou využiť pri príprave posterov. To ocenia najmä vedeckí pracovníci, ktorí neraz pripravujú práve prezentačné postery na rôzne vedecké podujatia. Ale takisto môžu byť využité napríklad pri príprave reklamných plagátov a podobne. Vynechať nemožno ani dobrú spoluprácu balíčka s dokumentmi triedy `beamer`, čiže elektronickými prezentáciami.

Poďakovanie. Príspevok vznikol s prispením grantu KEGA-019ŽU-4/2023 „Inovatívne učenie matematiky s podporou Open Source“, podporovaného Slovenskou kultúrno-edukačnou grantovou agentúrou.

Literatúra

[1] BLAŠKO, R.: *LaTeX nie je farba na maľovanie.* Zborník príspevkov medzinárodnej konferencie OSSConf 2010, 1.–4. júla 2010, Žilina, str. 43–52. ISBN 978-80-970457-0-8.

[2] BLAŠKO, R.: *LaTeX nie je farba na maľovanie, ale na písanie.* Zborník príspevkov medzinárodnej konferencie OSSConf 2011, 6.–9. júla 2011, Žilina, str. 223–236. ISBN 978-80-970457-1-5.

[3] KOPKA, H. – DALY, P., W.: *LaTeX – Podrobný průvodce*, Brno, Computer Press, 2004, ISBN 80-722-6973-9.

[4] KOZUBÍK, A.:*Naučím vás kresliť alebo predstavenie balíčka `Tikz`.* Zborník príspevkov medzinárodnej konferencie OSSConf 2012, 2.–4. júla 2012, Žilina, str. 91–96. ISBN 978-80-970457-2-2.

[5] RYBIČKA, J.: *LaTeX pro začátečníky*, Brno, KONVOJ 2003, Brno. ISBN 80-7302-049-1.

[6] SATRAPA, P.: LaTeX pro pragmatiky. `https://www.nti.tul.cz/~satrapa/docs/latex/`

[7] STURM, T., F.: *The `tcolorbox` package.* `https://ctan.gust.org.pl/tex-archive/macros/latex/contrib/tcolorbox/tcolorbox.pdf`.

Kontaktná adresa

RNDr. Aleš Kozubík, PhD., Katedra matematických metód, Fakulta riadenia a informatiky, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovenská Republika, *Aktuálna adresa*: SOIT, 010 01 Žilina, Slovenská Republika,

E-mailová adresa: `alesko@frcatel.fri.uniza.sk`

12. ročník Žilina 2. – 4. 7. 2024
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSS Conf

OSSConf 2024: 51–56

LATEX-OVÝ ROZMEROVNÍK

ALEŠ KOZUBÍK (SK)

Abstrakt. Úprava paramterov obrazca sadzby, odstavcov, zoznamov v texte, tabuliek, obrázkov a ich popisiek je bežnou súčasťou sadzby. Prakticky niet dokumentu, pri ktorom by sme sa nepotýkali s potrebou nejakým spôsobom upraviť implicitné prednastavené hodnoty. Pretože dokumentácia neobsahuje kompletný prehľadný zoznam dĺžkových registrov, je potrebné často zdĺhavo gúgliť potrebné rozmery, ktoré má používateľ regulovať. Tento príspevok je slovným úvodom k používaniu LATEX-ového rozmerovníka, ktorý je samostatnou prílohou tohto zborníka.

Kľúčové slová. LATEX, vzhľad stránky,dĺžkové registre.

LATEX MEASURE SHEET

Abstract. Editing of typeface parameters, paragraphs, in-text lists, tables, pictures and their labels is a common part of typesetting. There is practically no document in which we do not face the need to modify the implicit preset values in some way. Because the documentation does not contain a complete, comprehensive list of length registers, it is often necessary to google the suitable length registers that the user needs to adjust. This contribution is a verbal introduction to the use of the LATEX measure sheet, which is a separate independent appendix to this proceedings.

Keywords. LATEX, page layout, measure registers.

Úvod

Úprava sadzobného obrazca, čiže vzhľadu (alebo ak chcete layoutu) stránky je základom dobre spracovaného dokumentu. Tento prvok sadzby býva často problémom pre začiatočníkov, pri prechode od kancelárskych aplikácií typu „Office“ ku typografickému systému TEX/LATEX. Kým v kancelárskych balíkoch býva obraze sadzby prednastavený centricky na stránke s rovnomerne predvolenými okrajmi na všetkých stranách stránky, tak po kompilácii dokumentu v systéme TEX/LATEX je výsledný dokument, bez náležitých rozmerových úprav, umiestnený akosi excentricky, s „podivne“ nerovnomernými okrajmi. Navyše, obsah jednej stránky je potom príliš malý oproti očakávaniam. Má to však svoju logiku. Kým pri balíkoch typu „Office“, ako už názov napovedá, očakávame tvorbu kancelárskych výstupov, ktoré sú reprezentované predovšetkým listami. Sú teda prednastavené tak, aby bolo možné bez náročných manipulácií rýchlo napísať obchodný list, či jednoduché hlásenie alebo správu. Oproti tomu typografický

systém TEX/LATEX nahrádza prácu sadzača, kde výstupy môžu byť pomerne rôznorodé, počínajúc knihami, cez časopisy a noviny pokračujúc, až po listy alebo pri niektorých špeciálnych typoch publikácií končiac. Musí byť teda preddefinovaný tak, aby bol schopný zahrnúť do sadzby všetky prvky úpravy stránky, ako sú záhlavia, pätičky alebo marginálie. Pri okrajoch musí byť pamätané na väzbu knihy, čo spôsobuje nerovnomernosť okrajov na ľavej a pravej strane stránky. To sa prejaví aj odlišnou veľkosťou okrajov na párnych a nepárnych stránkach dokumentu, ak má byť tlačený obojstranne.

Takisto jednotlivé prvky samotnej sadzby je neraz potrebné rozmerovo upravovať. Či už sú to rozmery riadkov v tabuľkách alebo odstup obsahu jednotlivých buniek od deliacich čiar. Pri sadzbe do viacerých stĺpcov je potrebné nastaviť rozmery stĺpca, odstup medzi jednotlivými stĺpcami. Často sa regulujú rozmery zoznamov, ako je riadkovanie medzi jednotlivými položkami, veľkosť ich odsadenia od okraja bežného textu na stránke, veľkosť medzery medzi okolitým textom a zoznamom, a podobne.

V prostredí TEX/LATEX úpravu rozmerov riešime pomocou tzv. dĺžkových registrov, ktoré upravujú jednotlivé rozmerové parametre prvkov sadzby. Žiaľ, nikde som nenašiel nejako systematicky zoradené pomenovania jednotlivých registrov, často je ich potrebné zdĺhavo gúgliť, a neraz aj dlho rozmýšľať ako ich pomenovať, aby ste vygúglili naozaj ten rozmer, ktorý naozaj chcete. Účelom tohto príspevku je teda vytvoriť takýto základný zoznam rozmerových registrov, ktorý by používateľovi pomohol rýchlo sa zorientovať. Pri tom si autor nekladie nároky na úplnosť, ale zameriava sa na tie najfrekventovanejšie, s ktorými sa počas svojej praktickej činnosti so systémom TEX/LATEX stretol. Tento prehľad je pod názvom **LATEX-ový rozmerovník** vytlačený ako externý dokument a pre záujemcov je k dispozícii ako súčasť zborníka.

Základné informácie o rozmeroch a ich vzťahy k použitému písmu je možné nájsť v publikácii [7]. Detaily o rozvrhnutí sadzobného obrazca sa potom nachádzajú napríklad v [6], ktorá poslúži aj ako začiatočnícka učebnica TEX-u. O niečo podrobnejší výklad pravidiel prostredia TEX potom nájdete v [5]. Pre oboznámenie s prácou v prostredí TEX/LATEX je tiež možné využiť sériu článkov z našich predchádzajúcich ročníkov [1], [2] prácu s tabuľkami v článkoch [3] a [8].

1. Základné typografické rozmerové jednotky

Okrem bežných metrických jednotiek sa v typografii využíva aj celý rad iných pomocných rozmerových jednotiek. Ich popis nájdeme napríklad v [6]. Tu je prehľad všetkých prípustných jednotiek zhrnutý do tabuľky 1

Systém TEX/LATEX rozumie všetkým uvedeným rozmerovým jednotkám, je teda možné použiť ľubovoľné z nich. Odporúča sa, podľa možnosti maximálne využívať relatívne jednotky `ex` a `em`, ktoré nemajú pevnú hodnotu, ale sa prispôsobujú použitému fontu. Ak teda dôjde ku zmene základného fontu dokumentu,

Tabuľka 1. Základné typografické jednotky.

Rozmerové jednotky			
Značka	**Definícia**	**V bodoch**	**V milimetroch**
pt	bod = 1/72, 27 inch	1 pt	0, 35146 mm
mm	milimeter	2, 84 pt	1 mm
cm	centimeter	28, 4 pt	10 mm
in	palec	72 pt	25, 4 mm
ex	výška „x“	závisí od použitého fontu	
em	šírka „M“	závisí od použitého fontu	
bp	veľký bod = 1/72 inch	1, 00375 pt	0, 35278 mm
pc	pica	12 pt	4, 218 mm
dd	didot	1, 070 pt	0, 376 mm
cc	cicero (12 didot)	12, 84 pt	4, 512 mm
dn	new didot	1, 067 pt	0.375 mm
nc	new cicero (12 new didot)	12, 80 pt	4, 500 mm
sc	škálovaný bod	0, 000015 pt	0, 00000536 mm

tak si rozmery zachovávajú svoju proporcionalitu ku tomuto písmu, kým pri pevných jednotkách je táto proporcionalita samozrejme narušená.

2. Nastavenie dĺžkových registrov

Pre nastavenie hodnôt jednotlivých dĺžkových registrov máme k dispozícii dva základné príkazy:

```
\addtolength
\setlength
```

Príkaz `\addtolength{rozmer}{hodnota}` upraví veľkosť dĺžkového registra `rozmer` o veľkosť `hodnota`. Ako už bolo uvedené, `hodnota` môže byť zadané v ľubovoľnej dĺžkovej jednotke podľa tabuľky 1. Okrem toho môže byť vyjadrená aj pomocou už existujúcich dĺžkových registrov. Tak napríklad veľkosť zadaná ako `0.5\textwidth` predstavuje polovicu šírky textu. Pretože týmto príkazom upravujeme stávajúcu veľkosť daného rozmeru, môže `hodnota` nadobúdať aj záporné hodnoty, čím dôjde ku skráteniu daného rozmeru.

Príkazom `\setlength{rozmer}{hodnota}` potom nastavujeme príslušný rozmer na pevnú veľkosť `hodnota`. Aj tu môžeme použiť všetky akceptované dĺžkové jednotky podľa tabuľky 1, aj relatívne rozmery s využitím existujúcich dĺžkových registrov.

Občas sa stáva, že nepotrebujeme nastaviť veľkosť určitého rozmeru presnými metrickými jednotkami, ale chceme ju prispôsobiť veľkosti nejakého textu. V takom prípade môžeme použiť pomocné príkazy:

`\settowidth{rozmer}{text}`	Nastaví `rozmer` na šírku `text`.
`\settoheigth{rozmer}{text}`	Nastaví `rozmer` na výšku `text`.
`\settodepth{rozmer}{text}`	Nastaví `rozmer` na hĺbku`text`.

Pre pochopenie, čo sa rozumie pod výškou a hĺbkou textu odporúčame pozrieť napríklad [6]. Čiastočne je význam týchto pojmov naznačený aj v rozmerovníku.

Nezabudnime zmieniť ani možnosť definovať si svoj vlastný dĺžkový register príkazom:

```
\newlength{rozmer}
```

3. LaTeX-ový rozmerovník

Ak už vieme meniť hodnoty jednotlivých dĺžkových registrov, prichádza čas aj na úpravu vzhľadu stránky alebo jednotlivých komponentov sadzby. Tu je dôležité poznať názvy implementovaných dĺžkových registrov, aby sme úpravou toho správneho dosiahli požadovaný výsledok.

Prvým krokom je upravenie rozloženia stránky, k čomu slúži celkom 15 dĺžkových registrov. Význam jednotlivých rozmerov je spolu s rozložením prvkov stránky graficky ilustrovaný v rozmerovníku.

Druhým dôležitým krokom je nastavenie parametrov pre vzhľad jednotlivých odstavcov. Pomocou príslušných dĺžkových registrov nastavíme najmä dĺžku odsadenia prvého riadu odstavca od ľavého okraja, veľkosť vertikálnej medzery medzi jednotlivými odstavcami.

Významnou súčasťou dokumentov sú zoznamy, ktoré je zvykom odsadzovať od okrajov, či bežného textu, vertikálne odsadiť od textu odstavcov. Ako ukazuje ilustrácia v rozmerovníku, je tiež možné regulovať vertikálnu medzeru medzi jednotlivými položkami zoznamu, veľkosť návestí a ich odstup od položky zoznamu. Aj tieto rozmery sú, podobne ako rozloženie stránky, v rozmerovníku ilustrované graficky.

Veľmi často sa v dokumentoch stretávame s tzv. plávajúcimi prostrediami, čo sú predovšetkým tabuľky a obrázky. Aj nastavenie týchto prostredí disponuje škálou dĺžkových registrov, ktorými je možné upravovať ich umiestnenie vzhľadom na okolitý text. Sem patrí predovšetkým vertikálny odstup prostredia od okolitého textu, odstup popisu tabuľky či obrázku od samotného objektu ako aj od následného textu dokumentu. Tiež je možné upravovať maximálny podiel plávajúcich objektov z plochy stránky.

Tabuľky potom majú svoju vlastnú sadu dĺžkových registrov. Tie slúžia na úpravu umiestnenia obsahu jednotlivých buniek, teda najmä na reguláciu odstupu od deliacich čiar tabuľky, alebo úpravu výšky riadkov v tabuľke. Tak sa dosiahne

lepšia čitateľnosť obsahu tabuľky, keď pri implicitnom nastavení dochádza ku čiastočnému splynutiu okrajov buniek a ich obsahu.

Významnou súčasťou dokumentu, a silnou zbraňou systému TEX/LATEX, je sadzba v matematickom prostredí. Dĺžkové registre pre úpravu vzhľadu a umiestnenia matematických objektov tvoria poslednú časť v našom rozmerovníku.

4. Čo sa do rozmerovníka nezmestilo

Rozsah rozmerovníka bol limitovaný zámerom, aby sa celý zmestil na jeden list papiera formátu A4 a mohol tak slúžiť ak pohotová pomôcka v prípade potreby rýchlo nájsť názov dĺžkového registra, ktorý chceme upraviť. Občas sa stáva, že náš dokument spracúvame v podobe dvoch, alebo viacerých stĺpcov. Aj tu sa objavujú rozmery, ktoré je možné upravovať a meniť tak výsledný vzhľad dokumentu.

Sadzbu do viacerých stĺpcov umožňuje balíček `multicol`. Aj keď balíček umožňuje sadzbu prakticky do ľubovoľného počtu stĺpcov, nesmieme to s ich počtom preháňať. Čím je počet stĺpcov vyšší, tým obtiažnejšie sa stĺpce vyrovnávajú a vyvažujú a tomu zodpovedá aj výsledný vzhľad dokumentu. Je teda potrebné dbať na to, aby počet stĺpcov bol primeraný rozmerom stránky a veľkosti použitého písma.

Balíček `multicol` prináša aj nové rozmerové registre. Základom je veľkosť medzery medzi jednotlivými stĺpcami. Túto môžeme ovplyvniť určením hodnoty dĺžkového registra:

```
\columnsep
```

Stĺpce sa defaultne sádžu bez oddeľovacej zvislej čiary medzi nimi. Ak požadujeme zobrazenie čiary oddeľujúcej jednotlivé stĺpce, tak upravíme jej hrúbku zmenou hodnoty dĺžkového registra:

```
\columnseprule
```

Implicitná hodnota tohto registra je nastavená na 0pt, čím sa dosiahne jej neviditeľnosť. Môžeme zmeniť aj farebnosť tejto čiary, a to príkazom:

```
\def\columnseprulecolor{\color{farba}}
```

Hodnota `farba` potom zodpovedá pomenovaniu požadovanej farby.

Pre úplnosť dodajme, že šírka stĺpcov je v balíčku `multicol` počítaná podľa vzorca `(\linewidth-(n-1)*\columnsep)/n`, kde `n` je počet stĺpcov. V každom prípade majú všetky stĺpce rovnakú šírku, inak by vyvažovanie (čo je hlavným účelom balíčka `multicol`) nefungovalo.

Ako zaujímavosť spomeňme balíček `multicolrule`, ktorý umožňuje vkladať do dokumentu rôzne dekoratívne oddeľovače stĺpcov. Pri jeho použití sa odporúča sprístupniť aj balíček `tikz`, s ktorým veľmi dobre spolupracuje pri definovaní štýlov deliacej čiary.

5. Záver

Tento príspevok priniesol rozširujúci popis a návod na použitie LaTeX-ového rozmerovníka, ktorý je samostatnou prílohou zborníka. Autor si kládol za cieľ zaplniť medzeru v dokumentácii, a vytvoriť tak systematický prehľad dĺžkových registrov, užitočných pri príprave väčšiny bežných dokumentov. Veľkosť tohto zoznamu bola limitovaná rozsahom jedného listu papiera. Preto si nemôže klásť nárok na úplný, vyčerpávajúci zoznam. Ponúka však užitočnú pomôcku, s ktorou používateľ dokáže vyriešiť drvivú väčšinu situácií pri sadzbe.

Poďakovanie. Príspevok vznikol s prispením grantu KEGA-019ŽU-4/2023 „Inovatívne učenie matematiky s podporou Open Source“, podporovaného Slovenskou kultúrno-edukačnou grantovou agentúrou.

Literatúra

[1] BLAŠKO, R.: *LaTeX nie je farba na maľovanie*. Zborník príspevkov medzinárodnej konferencie OSSConf 2010, 1.–4. júla 2010, Žilina, str. 43–52. ISBN 978-80-970457-0-8.
[2] BLAŠKO, R.: *LaTeX nie je farba na maľovanie, ale na písanie*. Zborník príspevkov medzinárodnej konferencie OSSConf 2011, 6.–9. júla 2011, Žilina, str. 223–236. ISBN 978-80-970457-1-5.
[3] BLAŠKO, R.: *LaTeX a tabuľky*. Zborník príspevkov medzinárodnej konferencie OSSConf 2014, 2.–4. júla 2014, Žilina, str. 49–58. ISBN 978-80-970457-4-6.
[4] HAGEN, K.: *`multicolrule` – Decorative rules between columns*. `http://mirrors.ctan.org/macros/latex/contrib/multicolrule/multicolrule.pdf`
[5] KOPKA, H. – DALY, P. W.: *LaTeX – Podrobný průvodce*, Brno, Computer Press, 2004, ISBN 80-722-6973-9.
[6] RYBIČKA, J.: *LaTeX pro začátečníky*, Brno, KONVOJ 2003, Brno. ISBN 80-7302-049-1.
[7] RYBIČKA, J., ČAČKOVÁ, P., PŘICHYSTAL, J.: *Průvodce tvorbou dokumentů*, Bučovice, Nakladatelství Martin Stříž, 2011. ISBN 978-80-87106-43-3.
[8] RYBIČKA, J.: *LaTeX & tabulky & tipy & triky*. Zborník príspevkov medzinárodnej konferencie OSSConf 2014, 2.–4. júla 2014, Žilina, str. 139–151. ISBN 978-80-970457-4-6.

Kontaktná adresa

RNDr. Aleš Kozubík, PhD., Katedra matematických metód, Fakulta riadenia a informatiky, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovenská Republika, *Aktuálna adresa*: SOIT, 010 01 Žilina, Slovenská Republika,
E-mailová adresa: `alesko@frcatel.fri.uniza.sk`

12. ročník Žilina 2. – 4. 7. 2024
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSS Conf

OSSConf 2024: 57–68

JAK JSOU STŘEDOŠKOLÁCI VYBAVENI PRO TVORBU DOKUMENTŮ?

JIŘÍ RYBIČKA (CZ) A LUCIE PACÁKOVÁ (CZ)

Abstrakt. V letech 2022 a 2023 byl proveden dotazníkový průzkum u studentů nastupujících do prvního semestru na vysoké škole. Cílem bylo zjistit, do jaké míry je potřebné se věnovat ve výuce oblasti zpracování textů. Ze získaných dat vyplývá řada dílčích skutečností, souhrnně lze však konstatovat, že ve dvou ze tří sledovaných částí je výbava středoškoláků nedostatečná.

Klíčová slova. dotazníkové šetření, sebereflexe, jazykové znalosti, typografické zásady, technika dokumentů.

HOW ARE HIGH SCHOOL STUDENTS ABLE TO CREATE DOCUMENTS?

Abstract. In 2022 and 2023, a questionnaire survey was conducted among students entering their first semester at university. The goal was to find out to what extent it is necessary to devote attention to the area of text processing in teaching. A number of partial facts is obtained from the data, but in summary it can be stated that in two of the three monitored parts, the document skill of secondary school students is insufficient.

Keywords. questionnaire survey, self-reflection, language skills, typographic rules, document technique.

1. Úvod

Otázka položená v názvu příspěvku byla motivována především diskusí o náplni výuky předmětů situovaných v počátečních fázích univerzitního vzdělávání. Smyslem je připravit studenty na studium v ostatních předmětech, kde jsou vyžadovány různé semestrální práce, a pak také v závěru studia na bakalářskou práci.

V Rámcových vzdělávacích programech pro střední školy je zakotveno, že absolvent má být schopen „prezentovat výsledky své práce s využitím pokročilých funkcí aplikačního softwaru“ a příslušné učivo obsahuje „formy dokumentů a jejich struktura, zásady grafické a typografické úpravy dokumentu, estetické zásady publikování“ [2, s. 70].

Student opouštějící střední školu by podle těchto informací měl být schopen vytvořit dokument, jež je správný po stránce jazykové, typografické i technické. Z těchto předpokladů vyplývá, že na vysoké škole je zcela zbytečné vkládat do

obsahu jakéhokoliv předmětu učivo, které již student má znát, chtěli jsme proto zjistit, zda tomu tak skutečně je.

Studenti navštěvující první semestr svých studijních programů byli osloveni dotazníkem, jehož cílem bylo nejen zjistit, do jaké míry naplňují předpoklady dávané Rámcovým vzdělávacím programem, ale také, jak sami sebe v tomto ohledu vnímají. Další poněkud okrajovější informací bylo i zjišťování některých okolností výuky související s tvorbou dokumentů na nižších stupních škol.

2. Dotazníkové šetření

2.1. Organizace

O vyplnění dotazníku byli požádání studenti v prvním semestru svého studia, a to na Provozně ekonomické fakultě Mendelovy univerzity v Brně (dále jen PEF) a na Fakultě stavební Vysokého učení technického v Brně (dále jen FAST). Průzkum byl realizován na podzim v roce 2022 a na podzim v roce 2023.

Na PEF MENDELU byl dotazník realizován v univerzitním informačním systému, který je zároveň využíván pro testování a zkoušení studentů. Pro účely zpracování dat byly výsledky testů několika transformacemi z tohoto systému soustředěny do databázových tabulek.

Na VUT FAST byl pro dotazníkové šetření využit LMS Moodle. Výsledky byly získány v poněkud jiné podobě, bylo proto potřebné opět provést několik transformací a data pak soustředit do stejných tabulek jako z MENDELU.

Volba prostředí, v němž byly dotazníky předkládány respondentům, byla motivována zejména napojením na studijní systémy obou škol. Bylo tedy možné ještě před anonymizací ověřit, zda respondent odpovídá požadovanému výběru (český nebo slovenský rodilý mluvčí, první semestr studia na VŠ) a zda je jeho dotazník možné zahrnout do dále zpracovávaných dat.

2.2. Extrakce dat z dotazníků

Data pocházejí ze tří zdrojů: u Mendelovy univerzity z Univerzitního informačního systému (formát HTML), u VUT ze systému Moodle (formát PDF a JSON). Ke zpracování byly využity skripty v jazyce Python.

Pro zpracování dat z HTML stránek byla daná stránka s vyplněným testem stažena jako textový řetězec. Na základě analýzy stránek byly nalezeny tzv. patterny – řetezce, které jedinečně určují začátek hledaného textu, ať už identifikace studenta, znění otázky či odpovědi. Získaná data byla zapsána do CSV souboru a poté uložena do databáze.

Data z JSONu byla pro zpracování přístupnější už z principu zápisu formátu JSON. Zvláštností bylo, že jedno pole obsahovalo otázku i všechny možnosti, bylo tedy potřeba tyto dvě entity oddělit a všechny možnosti uložit pro pozdější zpracování. Další pole obsahovalo možnosti vybrané studentem, které se následně

zapsaly do CSV souboru jako vybrané, zatímco ostatní (nezvolené) možnosti se musely zapsat jako neoznačené.

Nejnáročnější ke zpracování byly PDF soubory – tento formát nebyl příznivý vůči automatickému zpracování, protože každý text měl jinou strukturu a převod souboru na text byl tak nejednoznačný. Věty byly rozdílně narušeny neznámými znaky, vybrané odpovědi byly těžko identifikovatelné, a právě proto bylo zvoleno částečně ruční zpracování – odpovědi studentů byly přepsány v jednodušším formátu do samostatného souboru. Následně byly zprácovány skriptem a opět zapsány do CSV souboru a uloženy do databáze.

2.3. Konstrukce dotazníku

Základní principy dotazníku vycházejí z konstrukce použité pro výzkum finanční gramotnosti prováděného na širokém vzorku studentů v letech 2015–2020 [1]. V souhlase s uvedeným konceptem byly otázky rozčleněny do tří částí – anamnestická, sebereflexní a znalostní.

V anamnestické části bylo pro nás důležité zjistit některé prvky, s nimiž se respondent setkal ve svém dosavadním studiu. Jde o počet předmětů, v nichž bylo obsaženo téma zpracování dokumentů, dále o používanou zpětnou vazbu od učitelů, tj. zda byli respondenti upozorňováni na formální nedostatky ve svých dokumentech, a o programové vybavení, s nímž dosud respondenti pracovali.

Sebereflexní část měla za cíl zjistit vlastní pohled respondenta na svou schopnost efektivně vytvářet bezchybné dokumenty. K této části pak byla jako komplement sestavena znalostní část, v níž byly otázky rozděleny na tři oblasti:

- Jazyková průprava – dvě poněkud rozsáhlejší otázky, v nichž respondenti rozhodovali o správnosti uvedených vět (frekventované jazykové chyby); dotazníky byly vytvořeny ve variantě pro češtinu a ve variantě pro slovenštinu pro respondenty přicházející ze Slovenska.
- Typografické zásady – pět otázek s nejzásadnějšími parametry sazby (písmo, odstavce, stránky).
- Technologická průprava – šest otázek reprezentujících optimální využití nejfrekventovanějších nástrojů v programu typu Word.

Na závěr dotazníku byla respondentům poskytnuta možnost se vyjádřit volným textem k jakémukoliv aspektu tohoto šetření.

3. Vybrané výsledky

Za oba dva roky dosud prováděného šetření byla získána validní data od 977 respondentů. Byly vyřazeny odpovědi, v nichž chybělo vyjádření na větší počet otázek, zahrnuty však byly testy, v nichž se mohla vyskytnout nějaká neodpovězená otázka. Proto počet odpovědí na jednotlivé otázky nemusí vždy dávat dohromady oněch 977.

Na tomto vzorku lze provést řadu analýz. Motivy výzkumu uvedené v úvodu příspěvku vedou k otázce, zda a s jakým rozsahem umístit do předmětů na vysoké škole témata zabývající se zpracováním textů.

Vybereme stručný přehled zajímavých výsledků ze znalostní části dotazníků. Jak už bylo uvedeno, jednalo se o otázky ze tří okruhů – jazykového, typografického a technického – které se podílejí na celkové schopnosti vytvářet textové dokumenty. Celkové výsledky se skládají z několika skupin respondentů, počty v jednotlivých skupinách přehledně uvádí tab. 1.

Tabulka 1. Počty respondentů v jednotlivých skupinách

Skupina	2023		2024	
	cz	sk	cz	sk
PEF	429	42	279	18
FAST	62	8	121	17

3.1. Jazykový okruh

Cílem otázek z jazykového okruhu bylo zjistit, jak jsou respondenti schopni rozpoznat jazykové (pravopisné) chyby v předložených větách. Jedna otázka byla zaměřena na jevy, které se většinou váží na použití v počítačových dokumentech, ale v ručně psaných textech mají nízký nebo nulový význam – pomlčky, uvozovky, mezerování interpunkce. Tato problematika je na nižších stupních škol vyučována pouze částečně, a to většinou v předmětech souvisejících s informatikou (ovládání softwaru pro zpracování textů), nikoliv ve výuce mateřského jazyka, kam tato problematika tematicky patří. Druhá otázka pak byla orientována na frekventované jazykové jevy podchycené ve standardní jazykové výuce, v nichž se podle zkušeností autorů příspěvku často objevují chyby.

V těchto dvou otázkách se lišily verze dotazníku pro české a slovenské respondenty. V otázce týkající se speciálních znaků a interpunkce jsme slovenskou verzi vytvořili velmi podobně jako českou, ovšem s respektováním specifik slovenského pravopisu. U druhé otázky jsme vycházeli z publikovaných informací o frekventovaných chybách [3] a navržený tvar jsme nechali prověřit rodilou mluvčí.

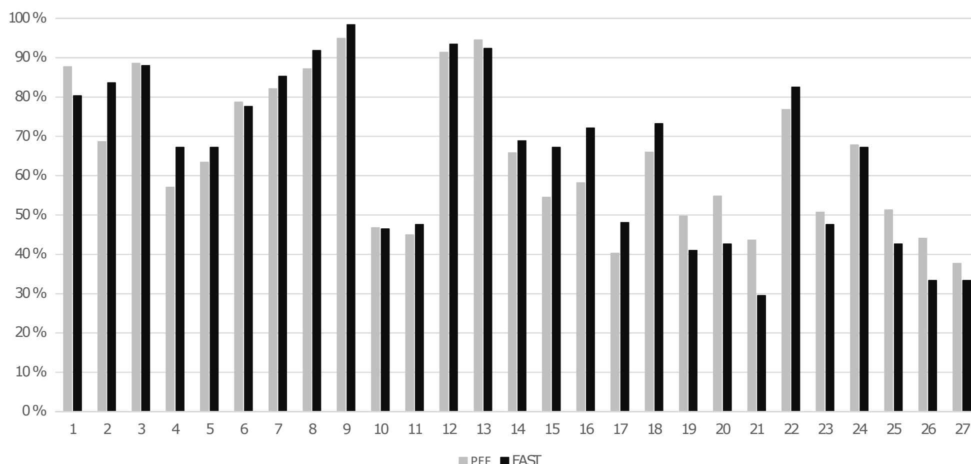
Tabulka 2. Úspěšnost jazykové otázky pro speciální znaky

CZ	2023		2024	
	n	p	n	p
	–	%	–	%
PEF	429	65,0	279	64,2
FAST	62	65,1	121	65,7

SK	2023		2024	
	n	p	n	p
	–	%	–	%
PEF	42	61,5	18	64,7
FAST	8	60,1	17	62,4

Oproti intuitivnímu předpokladu, že jazykové jevy ze standardní výuky budou mít větší úspěšnost než druhá skupina, jsou výsledky poměrně vyrovnané, jak ukazuje tab. 2. Výsledky slovenských respondentů nejsou ovšem z důvodu velmi malých počtů příliš použitelné.

Na obr. 1 jsou graficky souhrnně znázorněny úspěšnosti (v %) jednotlivých vět obsahujících různé speciální znaky, u nichž měli respondenti rozhodnout, zda jsou správně, nebo chybně. Graf obsahuje souhrn za oba roky. Je vidět, že úspěšnosti se příliš neliší mezi PEF a FAST.



Obrázek 1. Jazyk – speciální znaky; úspěšnost (%) pro jednotlivé věty (pořadová čísla 1–27); souhrn za oba sledované roky, varianta pro české respondenty

Na PEF bylo největším problémem posouzení věty s textem „Venkovní palubkové dveře s rámem 120x40 mm prosklené nejsou na skladě.“, která byla 441 respondenty prohlášena za správnou, byť je místo znaku „krát“ použit znak „x“ a navíc není správně mezerován. Na FAST klasifikovalo tuto větu správně pouhých 33 % respondentů a zajímavé je, že zcela stejnou úspěšnost měla na FAST i tatáž věta zapsaná správně. Nejhůře pak na FAST dopadla věta „Objednali jsme 14denní zájezd, ale ještě to nemáme potvrzeno.“, u níž pouze 29,5 % respondentů označilo její správnost. Přitom chybný tvar „14-denní“ odhalilo 82,5 % respondentů, avšak druhý chybný tvar „14-ti denní“ jen 47,5 %.

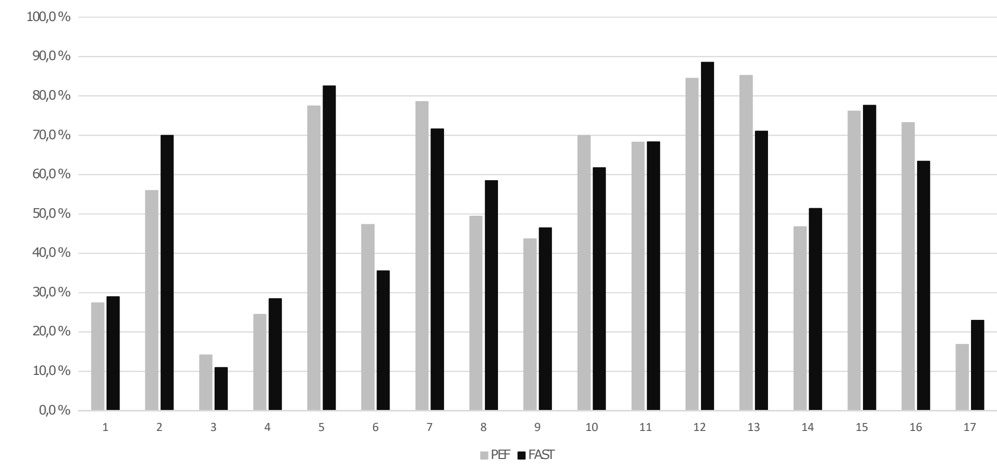
Druhá jazyková otázka obsahovala 17 vět v obou jazykových variantách, u nichž měli respondenti opět rozhodnout, zda jsou správné podle pravidel pravopisu. Zde se jazykové varianty liší více, protože repertoár frekventovaných chyb je rozdílný. Uvedeme opět celkové úspěšnosti a na grafu pak rozbor úspěšností jednotlivých vět (tab. 3, obr. 2).

Na obou školách byla jako nejméně úspěšná odhalena věta „Chtěl něco podotknout – asi se mu něco nezdálo –, ale nakonec mlčel.“ (14,1 %, resp. 10,9 %).

Tabulka 3. Úspěšnost jazykové otázky pro základní pravopis

CZ	2023		2024	
	n	p	n	p
	–	%	–	%
PEF	429	55,9	279	54,3
FAST	62	52,9	121	56,3

SK	2023		2024	
	n	p	n	p
	–	%	–	%
PEF	42	62,0	18	65,7
FAST	8	52,2	17	60,9



Obrázek 2. Jazyk – základní pravopis; úspěšnost (%) pro jednotlivé věty (pořadová čísla 1–17); souhrn za oba sledované roky, varianta pro české respondenty

Pravděpodobně se respondenti nechali zmást posloupností pomlčky a čárky. Druhá v pořadí byla opět shodně věta „Vyšel ze dveří, když v tom si vzpomněl na zapnuté světlo.“, kterou jako chybnou označilo pouze 16,8 %, resp. 23,0 % respondentů. Chybný výraz „v tom“ (správně má být příslovce „vtom“ = v tu chvíli, najednou) byl pravděpodobně kamenem úrazu podobně jako příslovce „namístě“ = správně, v pořádku), které v chybném tvaru „na místě“ bylo obsaženo ve větě č. 4.

3.2. Typografický okruh

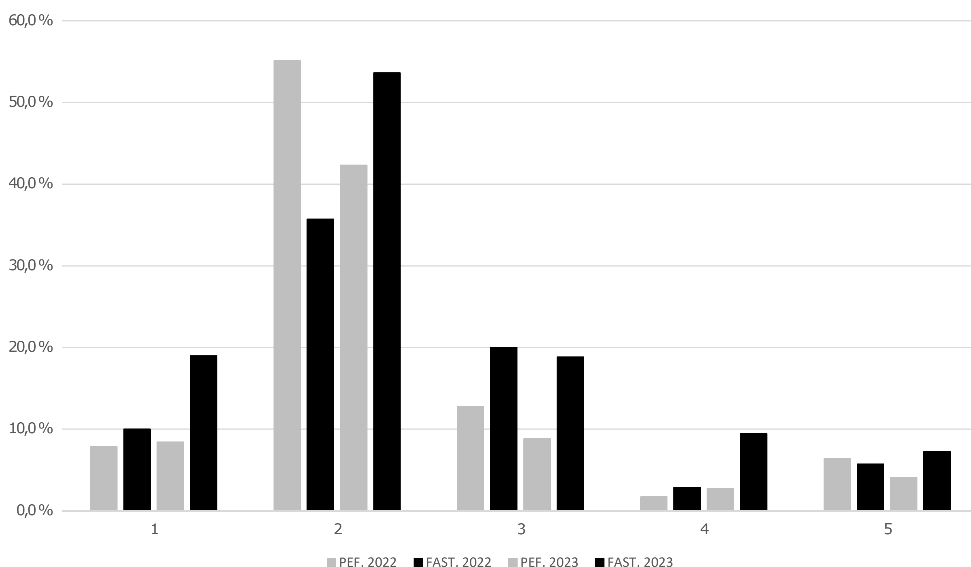
U pěti otázek zabývajících se vybranými typografickými zásadami bylo zajímavé sledovat, která pravidla se dostávají do všeobecného povědomí a která jsou naopak málo známá. Zde se ukazuje poměrně přesně, co ve výuce pravděpodobně chybí. Typografická pravidla byla jednotlivými otázkami zastoupena tak, aby byla zmíněna každá z následujících oblastí: volba vhodného základního písma, typografické měrné jednotky, sazba odstavců (zde byly dvě otázky – řádkování a odstavcová zarážka) a stránková úprava. Vždy jedna z nabízených možností byla

„Nevím.“, abychom také zjistili počet respondentů, kteří přiznají, že si s tímto parametrem nevědí rady.

V tab. 4 a v grafu na obr. 3 jsou zobrazeny celkové výsledky – úspěšnosti každé z pěti otázek souhrnně za českou i slovenskou skupinu respondentů. V tabulce jsou navíc upřesněny počty odpovědí – respondenti některé otázky výjimečně vynechali.

Tabulka 4. Úspěšnost otázek z oblasti typografických pravidel

	PEF, 2023		FAST, 2023		PEF, 2024		FAST, 2024	
	n	p	n	p	n	p	n	p
	–	%	–	%	–	%	–	%
Otázka 1	471	7,9	70	10,0	297	8,4	137	19,0
Otázka 2	470	55,1	70	35,7	293	42,3	138	53,6
Otázka 3	470	12,8	70	20,0	295	8,8	138	18,8
Otázka 4	470	1,7	70	2,9	293	2,7	138	9,4
Otázka 5	468	6,4	70	5,7	295	4,1	138	7,2



Obrázek 3. Úspěšnosti otázek z oblasti typografických pravidel; souhrn za oba roky

Nejlépe si respondenti poradili s otázkou č. 2: „V nejrůznějších pokynech pro formální úpravu dokumentů se lze dočíst: „Velikost písma 12.“ Co znamená zmíněná velikost 12?“. Z nabízených odpovědí zvolili průměrně ve 48,4 % (PEF) a 44,7 % (FAST) typografické body (za jednotlivé roky tyto výsledky mírně kolísají). Z jiných možností byla zhruba ve čtvrtině případů volena možnost „12

pixelů“ a o něco méně i „12 milimetrů“. Zhruba 10 % respondentů přiznalo odpověď „Nevím“.

U ostatních otázek jsou úspěšnosti výrazně nižší. Otázka č. 1 „Pojmem **řádkování** myslíme vzdálenost dvou po sobě jdoucích základních linek v běžném textu. Tuto vzdálenost podle typografických pravidel **nastavujeme**:“ měla souhrnnou úspěšnost za oba roky 8,1 % na PEF a 14,4 % na FAST. Zajímavý může být posun na PEF, ovšem nikoliv ve prospěch správné hodnoty. Zatímco v roce 2022 přes 30 % respondentů vybralo chybnou možnost „hodnotu 1,5“, v roce následujícím se podobný podíl přesunul k jiné chybné hodnotě „přesně 1,2“. Poměrně vysoký je podíl respondentů, kteří vybrali možnost „Nevím“ na FAST. V roce 2022 to bylo 30 %, o rok později ještě přes 23 %. Na PEF byla tato možnost vybrána v cca 16, resp. 13 procentech.

Druhý zásadní parametr odstavcové sazby, velikost odstavcové zarážky, představovala otázka č. 3. Možnost „Nevím“ zde volila čtvrtina až třetina respondentů, zhruba stejný podíl měla nesprávná možnost „vždy 1,25 cm“. K těmto dvěma velkým skupinám náleží ještě třetí s další nesprávnou volbou „5 úhozů, tj. polovina palce, přesně 1,27 cm“ v četnosti dosahující 20 %. Zbytek nepřesahující 20 % pak volil správnou možnost.

Ve čtvrté otázce měli respondenti zvolit optimální písmo pro dokument určený k tisku na papír. V každé možnosti byly nabízeny dva typy (názvy). Z více než poloviny volili respondenti PEF možnost dvou nepoužitelných písem „Times New Roman nebo Garamond“, na FAST tuto možnost volila jen zhruba čtvrtina. Dalších více než 40 % respondentů PEF volilo nesprávnou možnost „Arial nebo Calibri“, na FAST to bylo více než 60 %. Správnou možnost „Cambria nebo Constantia“ volilo mizivých několik procent (méně než 3), jen skupina na FAST v roce 2023 se poněkud vymyká s 9,4 %. Pozitivně lze hodnotit, že možnost „Comic Sans nebo Courier New“ nevolil téměř nikdo (jen 2 respondenti na PEF, 2023). V této otázce si byli respondenti velmi jisti, protože možnost „Nevím“ volila přibližně 2 %.

V poslední otázce bylo potřebné zvolit stránkové okraje. Správnou možnost respektující pozici optického středu stránky volily skupinky v četnosti od 4 do 7 procent. Nesprávnou možnost se všemi okraji 2,5 cm volilo téměř vyrovnaně cca 36 %, druhou nesprávnou možnost s okraji nahoře a dole 2,5 cm a vlevo a vpravo 2 cm dokonce kolem 51 %. Zajímavé je, že podobnou možnost, tj. nahoře a dole 1,27 cm a vlevo a vpravo od 2 do 2,5 cm, volilo přibližně stejně málo respondentů jako možnost správnou, tedy nejvýše 7 %.

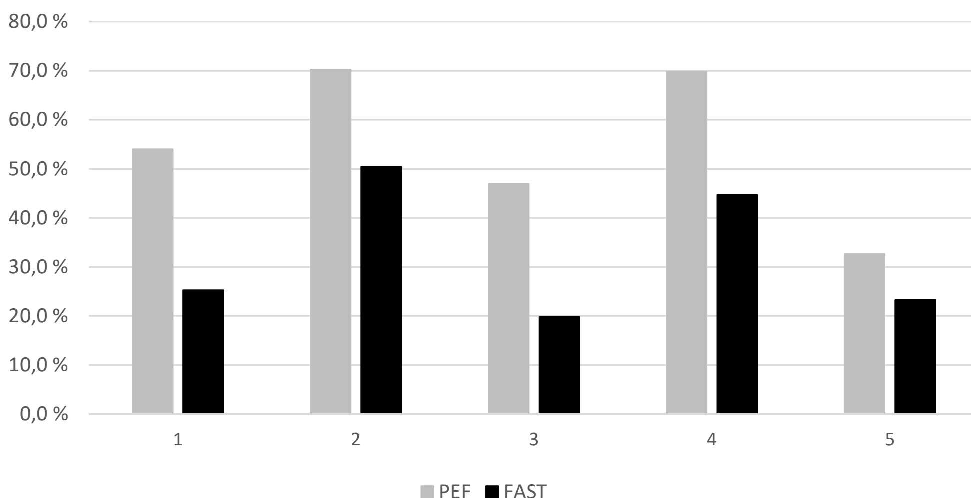
3.3. Technický okruh

Otázky týkající se realizace dokumentu v konkrétním programovém vybavení byly částečně koncipovány jako doplněk k otázkám typografického okruhu. Při přípravě této části jsme stáli před obtížnou volbou, jak otázky formulovat, aby respondenti přesně pochopili, jaké konkrétní funkce máme na mysli. Po dlouhé

diskusi jsme se rozhodli, že se budeme odvolávat na služby a funkce programu Word, protože jsme se odvážili předpokládat, že se s tímto programem (a jedině s tímto) již všichni respondenti setkali a použili jej. Tento předpoklad se téměř splnil, z celkových 977 odpovědí se plných 957 vyjádřilo, že tento program již použili. Ve stejném duchu je vytvořena i česká norma ČSN 01 6910 [4], která hovoří o „funkcích textového procesoru“, ale z názvů těchto funkcí jasně vyplývá, že jde právě o program Word.

V pěti otázkách byly představeny vybrané frekventované situace a respondenti byli požádáni, aby z nabízených řešení vybrali *optimální* variantu. V šesté otázce, která se zabývala tvorbou matematického výrazu, nešlo o optimální variantu, ale o správnou konstrukci, proto sem byla doplněna možnost „Nevím“. O každé možnosti respondenti rozhodovali, zda je správná, nebo špatná. Počty odpovědí v souhrnu za všechny varianty tedy mohou být vyšší, než je celkový počet respondentů.

Úspěšnosti prvních pěti otázek ve srovnání mezi PEF a FAST ukazuje graf na obr. 4. Ve všech pěti otázkách byli studenti PEF lepší (i výrazně) než studenti FAST.



Obrázek 4. Úspěšnosti otázek z oblasti použité technologie; souhrn za oba roky

První otázka se týkala zahájení nového odstavce a jeho zarážky. Z nabízených možností pouze jedna představovala optimální postup, tj. pouhý stisk klávesy Enter, přičemž hodnota odstavcové zarážky je určena parametrem stylu. Z nesprávných možností byla nejatraktivnější varianta s klávesou Enter a následním stiskem klávesy Tab, která na začátku nového odstavce má vytvořit odstavcovou zarážku. Tu volili studenti PEF v cca 80 %, studenti FAST pak v 60–70 %.

Druhá otázka prezentovala situaci, kdy je potřebné spojit titul a jméno osoby, kdy technickým řešením je vložení nezlomitelné mezery. Tuto variantu by použilo na PEF 74,1 % (2022), resp. 66,3 % (2023) respondentů, zatímco na FAST jen 44,3 %, resp. 56,5 %. Velmi frekventovaná byla však na PEF také zcela chybná možnost vložení umělého konce řádku před titul – až 70 %. Na FAST podlehlo této chybě pouhých 30 % respondentů.

Podobnou situací, avšak ve vertikálním směru, se zabývala třetí otázka – nadpis vyskytující se na konci stránky. Masivně byla volena nesprávná možnost umístění nepodmíněného konce stránky před nadpis – na PEF přes 80 %, na FAST „jen“ 40–50 %.

Způsobem sazby běžného hladkého textu v dokumentu s výstupem na papír se zabývala čtvrtá otázka. Vedle optimální možnosti sazby do bloku se zapnutým dělením slov byla také často volena možnost sazby se zarovnáním vlevo a bez dělení slov (PEF 58 a 63 %, FAST 51 a 36 %).

Pátá otázka směřovala k použití stylů – technologicky velmi důležitý prvek efektivní úpravy dokumentů. Na PEF by tuto cestu zvolilo jen 35,7 %, resp. 29,6 %, na FAST v roce 2023 dokonce jen 5,1 %. Laická varianta s kliknutím do pásu karet nebo použitím identické klávesové zkratky byla naopak volena ve více než 70 % na PEF a kolem 50 % na FAST.

V šesté otázce zabývající se konstrukcí matematického výrazu e^{3x} bylo zjišťováno, jak se respondenti vyrovnají s faktem, že konstanta „e“ má být sázena obyčejným řezem stejně jako trojka v exponentu, zatímco proměnná x musí být matematickou kurzívou. Použitelné možnosti byly voleny na PEF v přibližně 40 %, na FAST mezi 12 a 29 %. Možnost „Nevím“ byla volena od 10 do 30 %.

3.4. Otevřená otázka

Součástí dotazníku byla i otázka, která umožňovala respondentům vyjádřit se k průběhu dotazníku či dodat nějaké připomínky k oblasti zpracování textů. Stojí za zmínku, že třetina respondentů uvedla, že považují tento dotazník za užitečný, že je inspiroval k dostudování pravopisných i typografických jevů a že typografie i pravopis jsou důležitou oblastí pro srozumitelné dorozumívání mezi lidmi. Necelých 15 % respondentů se výslovně vyjádřilo, že by v této oblasti uvítali další vzdělávání, ať už samostatné, nebo v rámci odpovídajících předmětů studijních plánů základních a středních škol, nebo právě zde na vysoké škole.

4. Diskuse

Z uvedených výsledků můžeme odvodit některé zajímavé informace. Oproti našemu předpokladu nebyl zjištěn podstatný rozdíl mezi znalostmi speciálních znaků s jejich mezerováním a dalšími jazykovými jevy. Všeobecně lze také konstatovat, že schopnosti detekovat běžné jazykové chyby nejsou špatné, i když samozřejmě i zde jsou značné rezervy.

Překvapením v opačném směru byly zjištěné znalosti typografických pravidel. Zde se objevuje značný prostor pro jejich umístění do výuky. Ukazuje se, že do všeobecného povědomí zdaleka nepronikla informace o vhodných písmech (z nabízených vhodných typů se písmo Cambria a Constantia v systému nachází již 18 let), o optimálním řádkování a dalších parametrech odstavcové sazby (stále přežívají různé zcestné a nefunkční rádoby pokyny pro úpravu různých prací, kde se neustále omílá hrubé nepochopení reality se strojopisným řádkováním 1,5 a s nepoužitelným písmem Times New Roman) a o stránkových okrajích respektujících pozici optického středu stránkové plochy. Řada těchto nesmyslů vyplývá i z chaoticky přednastavených hodnot v nejmasověji používaném amatérském a bezkoncepčním nástroji – MS Wordu.

Oblast schopností využít technologických prvků nejpoužívanějšího softwaru rovněž nevykazuje nijak oslnivé výsledky. Zejména využití stylů je na zcela nedostatečné úrovni, podobně i prvků potřebných pro téměř každý dokument – vhodnou úpravu odstavců a vazby prvků tak, aby nevznikaly běžné chyby (nezlomitelné mezery, vazba odstavců při stránkovém zlomu). Pro každou technologii použitou pro zpracování dokumentů platí, že musí splňovat implementaci dvou hlavních typografických zásad – zásady jednotnosti (významově stejné prvky mají mít identický vzhled) a zásady kontrastu (významově odlišné prvky se mají vzhledově odpovídajícím způsobem dostatečně lišit). To má být dosaženo efektivním způsobem, tedy nástroji, které automaticky při každé úpravě a změně zajistí konzistenci obou těchto zásad. Proto je použití odpovídajících stylů zcela neopominutelnou součástí technologie zpracování textů ve Wordu a podobných nástrojích, přičemž z typografického hlediska nelze akceptovat značnou část předdefinovaných stylů a je nezbytné používat styly vlastní. Lze konstatovat, že i technologickou oblast se zmíněným obsahem je vhodné zařadit do výuky, a prezentovat tak možnost efektivní práce s dokumentem.

Samotným zařazením do výuky však není problém dostatečně vyřešen. Současně s tím musí existovat dostatečně silná zpětná vazba – vytvoří-li student jakýkoliv dokument (seminární práce v libovolném předmětu, závěrečná práce), musí dostat jednoznačnou informaci od hodnotitele, zda jsou základní kritéria formální stránky splněna. Je naprosto nepřípustné, aby hodnotitelé z vlastní neznalosti ignorovali hrubé nedostatky typografické úpravy a dokonce se i za této situace vyjadřovali kladně v kritériu „Formální úprava“ u závěrečných prací.

5. Závěr

Provedené dotazníkové šetření poskytlo velké množství údajů, z nichž byly vybrány pouze některé aspekty, na jejichž základě je možné usoudit, zda a do jaké míry je vhodné umístit oblast zpracování textů do výuky v počátečních fázích vysokoškolského studia.

Ukazuje se, že zejména v oblasti typografických zásad a také v některých částech technologické implementace je tato výuka nezbytná. Následně je potřebné řešit, jakými formami a v jakém kontextu je možné dosáhnout identifikovaných cílů. Odpověď na otázku položenou v názvu příspěvku totiž lze formulovat jako „nedostatečně“.

Konstrukce dotazníků, některé výsledky, zpětná vazba od respondentů a vlastní poznatky autorů také ukazují, že tento výzkum by mohl pokračovat i v následujících letech, ale s určitými drobnějšími úpravami. Přínosem by bezpochyby mohla být aplikace dotazníku na dalších školách s možností srovnání nejen na národní, ale i na mezinárodní úrovni, jako tomu bylo i v případě zmíněného výzkumu finanční gramotnosti (Slovensko – Česko).

Reference

[1] Kozubikova, Z. (2016). Financial Literacy in Selected Groups of Students. In P. Slavíčková (Ed.), *Proceedings of the Conference: Knowledge for Market Use 2016: Our Interconnected and Divided World*, Olomouc: Societas Scientiarum Olomucensis II, pp. 222–230.

[2] MŠMT (2021). *Rámcový vzdělávací program pro gymnázia, RVP G* [online]. Dostupné na `https://www.edu.cz/wp-content/uploads/2021/09/001_RVP_GYM_-vyznacene_zmeny.pdf`

[3] Koníčková, J. *Najčastejšie gramatické chyby v slovenčine. Robíte ich aj vy?* [online] [vid. 20. 4. 2022] Dostupné na `https://eduworld.sk/cd/jaroslava-konickova/3606/najcastejsie-gramaticke-chyby-v-slovencine-robite-ich-aj-vy`

[4] ÚNMZ. (2014). ČSN 01 6910 – Úprava dokumentů zpracovaných textovými procesory. Praha.

Kontaktní adresy

doc. Ing. Jiří Rybička, Dr., Ústav informatiky, Provozně ekonomická fakulta, Mendelova univerzita v Brně, Česká republika,
E-mailová adresa: `rybicka@mendelu.cz`, `https://user.mendelu.cz/rybicka`

Lucie Pacáková, Ústav informatiky, Provozně ekonomická fakulta, Mendelova univerzita v Brně, Česká republika,
E-mailová adresa: `pacakovalu@email.cz`

12. ročník Žilina 2. – 4. 7. 2024
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
osscconf.soit.sk
OSS Conf

OSSConf 2024: 69–76

VÝUČBA OPEN SOURCE DATABÁZOVÉHO SYSTÉMU V GEOGRAFICKÝCH INFORMAČNÝCH SYSTÉMOCH PRE ODBOR GEODÉZIA A KARTOGRAFIA

RÓBERT SÁSIK (SK), DÁŠA SMRČKOVÁ (SK), JAKUB CHROMČÁK (SK) A JANA IŽOVLTOVÁ (SK)

Abstrakt. Tento článok skúma výučbu Open Source databázových systémov v geografických informačných systémoch pre študentov odboru geodézia a kartografia. Zdôrazňuje význam spracovania údajov do informácií a využívanie priestorových databáz. Predstavuje postup inštalácie a konfigurácie PostgreSQL a jeho rozšírenia PostGIS pre spracovanie geografických dát.

Kľúčové slová. GIS, Open Source databázové systémy, PostGIS.

TEACHING OPEN-SOURCE DATABASE SYSTEM IN GEOGRAPHIC INFORMATION SYSTEMS FOR THE GEODESY AND CARTOGRAPHY STUDY PROGRAMME

Abstract. This article examines the teaching of Open Source database systems in geographic information systems (GIS) for students in the field of geodesy and cartography. It emphasizes the importance of processing data into information and the use of spatial databases. It presents the procedure for installing and configuring PostgreSQL and its extension PostGIS for processing geographic data.

Keywords. GIS, Open Source database systems, PostGIS.

Úvod

Dáta sú nespracované fakty, ktoré až spracovaním dostanú požadovanú štruktúru a význam a stávajú sa z nich informácie. Napr. číslo 1750 je údaj. Pokiaľ na základe informačného systému vieme, že údaj 1750 znamená napr. výmeru parcely č. 520, ide už o informáciu [2].

Databáza je zdieľaná integrovaná počítačová štruktúra, ktorá zahŕňa dáta a metadáta (dáta o dátach). Predstavuje množinu vzájomne súvisiacich dát uloženú na pamäťovom médiu, usporiadanú a organizovanú tak, aby podporila vykonávanie špecifických požiadaviek. Databázy sú ukladané v špeciálnych súboroch operačného systému a tvoria jadro celého databázového systému [1].

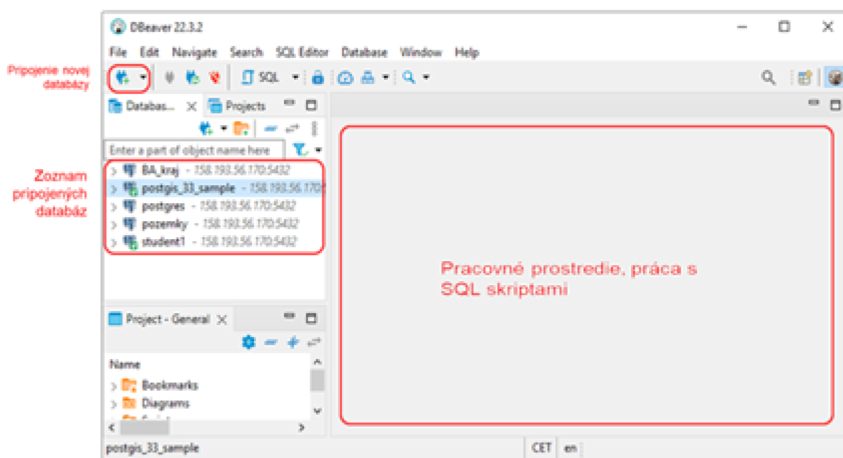
V rôznych oblastiach existuje potreba spravovať geometrické, geografické alebo priestorové údaje, čo znamená údaje spojené s priestorom. Priestor záujmu môže byť napríklad dvojrozmerná abstrakcia (častí) povrchu Zeme, čo je geografický priestor [3].

Preto vznikli tzv. priestorové databázy, ktoré sa najčastejšie spájajú s geografickými informačnými systémami. Priestorová databáza je databáza, ktorá je vylepšená na ukladanie a prístup k priestorovým údajom alebo údajom, ktoré vymedzujú geometrický priestor. Tieto údaje sú často spojené s geografickými polohami a prvkami alebo s vytvorenými prvkami, ako sú napr. mestá. Údaje o priestorových databázach sa ukladajú ako súradnice, body, čiary, polygóny a topológia.

1. Nástroje na prácu s GIS databázou

Pre potreby výučby predmetu Databázové systémy v GIS využívame databázový systém `PostgreSGL` s nadstavbou `PostGIS`, ktorý je nainštalovaný na katedrovom serveri s doménou `dbgis.kgd.uniza.sk`.

`PostGIS` je rozšírenie pre databázový systém PostgreSQL, ktoré umožňuje spracovávať geografické (GIS) dáta. Toto rozšírenie poskytuje databázové funkcie na prácu s geografickými objektami, ako sú body, čiary, polygóny a umožňuje vykonávať rôzne geografické operácie, ako sú zisťovanie vzdialenosti, zlúčenie geometrických objektov alebo vyhľadávanie objektov vo zvolenom geografickom rozsahu. `PostGIS` je populárnym nástrojom v oblasti geografického informačného systému (GIS) a je často používaný na ukladanie, spracovanie a analýzu priestorových dát v aplikáciách ako sú mapové služby, analytické nástroje alebo geografické aplikácie.



Obr. 1. Prostredie DBeaver na prácu s GIS databázami

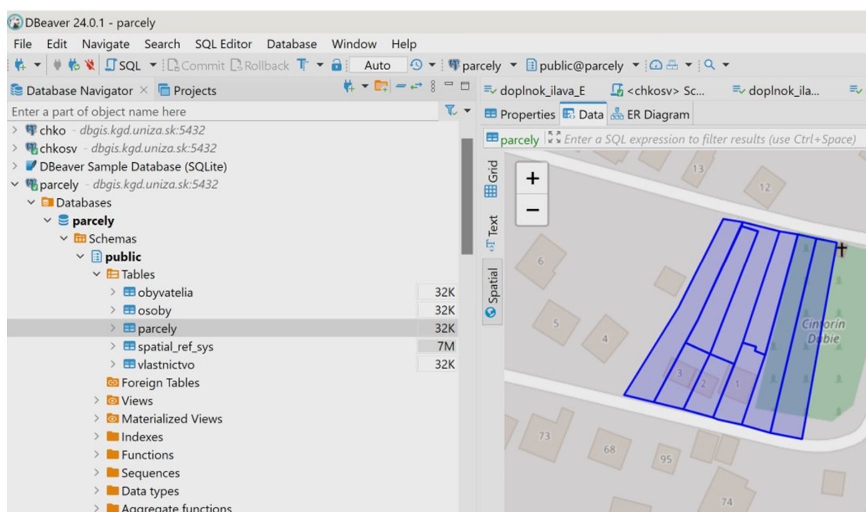
Na prácu s databázami v GIS pre výučbu študentov používame grafické rozhranie `DBeaver` a `QGIS`.

DBeaver je multi-platformový nástroj na správu databáz pre vývojárov, databázových administrátorov a analytikov. Je to Open Source nástroj, ktorý poskytuje užívateľské rozhranie pre pripojenie k rôznym typom databázových systémov vrátane PostgreSQL, MySQL, Oracle, Microsoft SQL Server a mnohých ďalších. DBeaver umožňuje užívateľom prehliadať štruktúru databázy, vykonávať SQL dotazy, vytvárať a upravovať tabuľky, indexy, uložené procedúry a iné objekty databázy. Okrem toho poskytuje nástroje na export a import dát, správu užívateľov a práv a podporuje rôzne formáty dátových súborov. Je to veľmi obľúbený nástroj v komunite vývojárov a databázových profesionálov pre svoju flexibilitu, bohatú funkcionalitu a schopnosť práce s rôznymi typmi databázových systémov.

Pre prácu s priestorovými dátami v prostredí PostgreSQL je potrebné aktivovať modul PostGIS, a to tak, že v skripte SQL zadáme príkaz

```
CREATE EXTENSION postgis
```

Pre praktické príklady v študentských prácach používame databázy parcely, Vlastnictvo_parciel, dbgis.



Obr. 2. Zobrazovanie priestorových dát priamo v prehliadači prostredia DBeaver

2. Praktické príklady prepojenia databáz s programom QGIS

Úloha A.

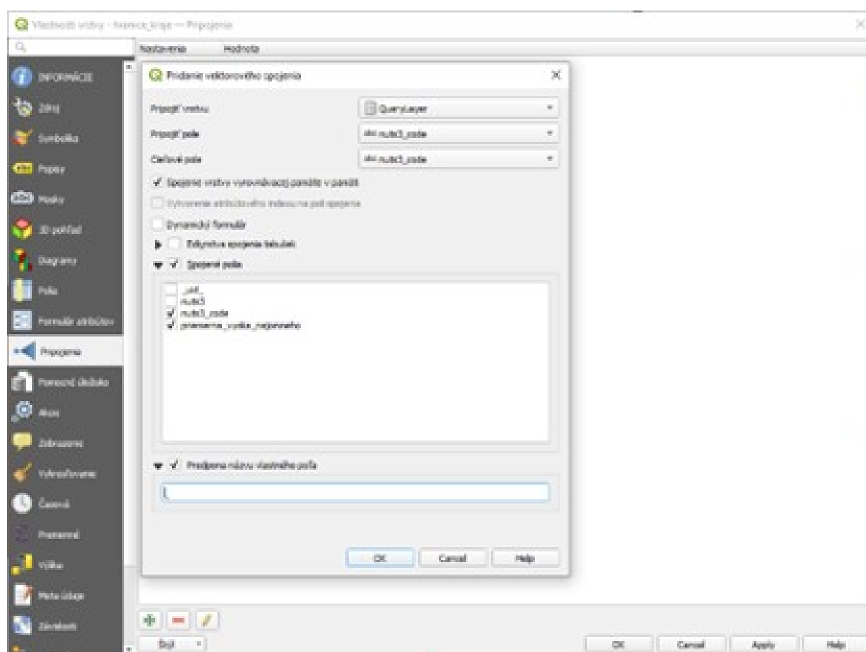
Zistite priemernú výšku nájomného poľnohospodárskej pôdy pre každý samosprávny kraj. Pomocou SQL príkazu vytvorte tabuľku s novým atribútom priemerna_vyska_najomneho. V QGIS-e pripojte tabuľku s priemernými výškami nájomného k existujúcim hraniciam samosprávnych krajov. Výsledok vyexportujte na

*disk PC do ľubovoľného formátu napr. `ESRI SHP`. Použite databázu **`dbgis`**, tabuľky **`vyska_najomneho_pp`**, **`hranice_kraje`**.*

Riešenie.
V programe `QGIS` sa pripojte na `PostgreSQL` databázu `dbgis`. Následne je potrebné otvoriť v programe `QGIS` nástroj `DB manager`. V databáze `dbgis` prejsť do schémy `public`. V okne nástroja `DB manager` zadať a spustiť `SQL` dopyt:

```
SELECT NUTS3, NUTS3_CODE, AVG(OVN2021)
AS priemerna_vyska_najomneho FROM vyska_najomneho_pp
GROUP BY NUTS3, NUTS3_CODE;
```

Výsledok `SQL` dopytu je možné načítať v `QGIS`-e ako novú vrstvu napr. pod názvom `QueryLayer`. Do prostredia `QGIS`-u načítať novú vektorovú vrstvu `hranice_kraje` a otvorte vlastnosti tejto vrstvy. Teraz je potrebné prepojiť výsledok dopytu (`QueryLayer`) s vrstvou `hranice_kraje` pomocou funkcie `join` (pripojenia), ktorá sa nachádza vo vlastnostiach vrstvy. Primárny kľúč nastaviť na `nuts3_code`.



Obr. 3. Spájanie atribútových tabuliek pomocou primárnych kľúčov

Úloha B.

Vytvorte tabuľku s názvom `vinice` z existujúcej tabuľky `sk_druhy_pozemkov`, kde zistite plochu viníc v m^2 pre každý okres. V tejto novej tabuľke vinice budú dva atribúty `kod_okresu` a `plocha_vinic_m2`. Spojte tabuľku `hranice_okresy`, v ktorej sú mapové dáta hraníc okresov SR s tabuľkou `vinice` a vytvorte z nej novú tabuľku s názvom `okresy_vinice`, ktorú zobrazíte v programe `QGIS`. (Každý študent pomenuje tabuľku napr. `okresy_vinice_rs`).

V tabuľke `okresy_vinice` vytvorte nový atribút `plocha_vinic_ha` a do neho vložte prepočítané m^2 na hektáre. V `QGIS`-e farebne kategorizujte okresy podľa veľkosti plôch viníc v ha. Použite databázu `dbgis`, tabuľky `sk_druhy_pozemkov`, `hranice_okresy`.

Riešenie.

V programe `QGIS` sa pripojte na `PostgreSQL` databázu `dbgis`, otvorte `DBmanager`. V okne nástroja `DB manager` postupne spustite `SQL` dopyty:

```
CREATE TABLE vinice AS SELECT kod_okresu
SUM(vinice_m2) AS plocha_vinic_m2
FROM sk_druhy_pozemkov GROUP BY kod_okresu;
```

```
CREATE TABLE okresy_vinice AS
SELECT *
FROM vinice
JOIN hranice_okresy ON hranice_okresy.idn3 = vinice.kod_okresu;
```

```
– Vytvorenie nového atribútu plocha_vinic_ha v tabuľke okresy_vinice
ALTER TABLE okresy_vinice
ADD COLUMN plocha_vinic_ha numeric;
– Aktualizácia nového atribútu pomocou prepočtu z m2 na ha
UPDATE okresy_vinice
SET plocha_vinic_ha = plocha_vinic_m2 / 10000;
```

V `QGIS`-e je potrebné načítať databázovú tabuľku `okresy_vinice` ako novú vrstvu. V nastaveniach vrstvy prejsť na možnosť `Symbology`, kde je potrebné nastaviť možnosť `Graduated`. Atribút, na základe ktorého sa robí symbolika je `plocha_vinic_ha`. Vyberie sa vhodná farebná schéma, na základe ktorej sa urobí klasifikácia triedy zobrazených hodnôt. Výsledná klasifikácia plochy viníc podľa okresov by mala vyzerať ako na obrázku 4.

Úloha C.

Vyberte všetky slovenské vrchy, ktoré majú nad 2000 m n.m. vrátane. Taktiež zobrazte ich názvy pomocou atribútu `nazvysb`. Zoraďte ich od najnižšieho po



Obr. 4. Plochy viníc podľa okresov

najvyšší. Vyselektujte atribút s geografickými údajmi. V `QGIS`-e vytvorte so symbolikou bodovej vrstvy a názvami vrchov. Zvoľte vhodnú podkladovú mapu napr. s použitím `WMS`.

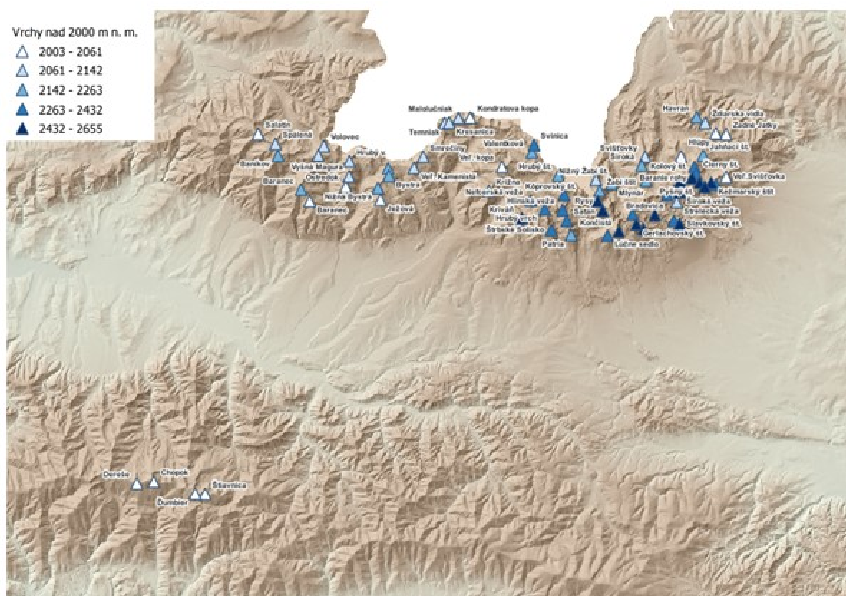
Riešenie.
V `QGIS`-e sa pripojte na `PostgreSQL` databázu `dbgis`. Následne v okne nástroja `DB manager` spustite `SQL` dopyt.

```
SELECT geom, nazvysb, vyska
FROM vyskove_body
WHERE vyska >= 2000 AND nazvysb IS NOT NULL
and nazvysb != ” ORDER BY vyska ASC;
```

Výsledok `SQL` dopytu načítajte do `QGIS`-u ako novú vrstvu. Potom novú vrstvu vhodne pomenujte, napr. `Vrchy nad 2000 m n.m.` a v nastaveniach vrstvy nastavte vhodnú symboliku. Pripojte novú webovú mapovú službu napr. z webovej stránky `https://www.geoportal.sk` službu `DMR~3.5`.

Nastavte vhodné usporiadanie vrstiev a ich vizualizáciu. Vo vrstve `DMR3` nastavte transparentnosť vrstvy.

Výsledná mapa by mala vyzerať ako na obrázku 5, ktorá bola vytvorená pomocou nástroja `Project>New Print Layer` v `QGIS`-e.



Obr. 5. Mapa výškových bodov nad 2000 m n.m.

3. Záver

Výučba databázových systémov v geografických informačných systémoch (GIS) pre odbor geodézia a kartografia prináša študentom významné výhody, keďže sa učia pracovať v Open Source aplikáciách, ako sú `PostgreSQL` s `PostGIS`, `DBeaver` a `QGIS`. Tieto programy sú voľne dostupné, čo umožňuje študentom trénovať si príklady z cvičení aj mimo univerzitných priestoroch, čo je veľkou výhodou v porovnaní s komerčnými softvérmi, ktoré často vyžadujú drahé licencie.

Používanie Open Source riešení podporuje inováciu a spoluprácu, čím sa zvyšuje kvalita vzdelávania a prípravy na profesionálnu kariéru. Tieto nástroje sú neustále vyvíjané a vylepšované širokou komunitou vývojárov a používateľov z celého sveta, čo zaručuje ich aktuálnosť a vysokú funkčnosť. Študenti tak získavajú praktické skúsenosti s modernými nástrojmi, ktoré sú široko využívané v priemysle. Naučia sa, ako efektívne spravovať a analyzovať veľké množstvo geografických dát, čo je kľúčová zručnosť v súčasnej dobe, keď sa množstvo dostupných dát neustále zvyšuje.

Integrácia týchto systémov do výučby je nevyhnutná pre rozvoj praktických schopností a technickej pripravenosti budúcich odborníkov v oblasti geodézie a kartografie. Študenti sa učia nielen teoretické základy databázových systémov a GIS, ale aj praktické príklady potrebné na riešenie reálnych problémov. Práca s `PostgreSQL` databázovým systémom spolu s nadstavbou `PostGIS` ich pripravuje

na využívanie priestorových databáz, ktoré sú základom mnohých GIS aplikácií, či už komerčných alebo nekomerčných.

Týmto spôsobom sa zvyšuje konkurencieschopnosť študentov na trhu práce a pripravujú sa na úspešnú kariéru v dynamicky sa rozvíjajúcom IT odbore, keďže moderná geodézia a kartografia spolupracuje aj s IT technológiami. Výučba databázových systémov v GIS teda predstavuje nielen prínos pre akademický rozvoj študentov v odbore geodézia, ale aj významný krok k ich budúcej profesionálnej úspešnosti.

Literatúra

[1] DELIKÁT, T.: *Základy databázových systémov*, Vydavateľstvo Delint, 2006, 209 s. ISBN 809694844X.

[2] ĎURAČIOVÁ, R.: *Databázové systémy v GIS*, Vydavateľstvo STU, 2014, 178 s. ISBN 9788022742924.

[3] GÜTING, H. R.: *An Introduction to Spatial Database Systems*, Praktische Informatik IV, Fern Universität Hagen, Germany, *Special Issue on Spatial Database Systems*, VLDB Journal (Vol. 3, No. 4, October 1994), [cit. 20.5.2024], `https://www.fernuni-hagen.de/pi4pp/papers/IntroSpatialDBMS.pdf`.

Kontaktné adresy

Ing. Róbert Sásik, PhD., Katedra geodézie, Stavebná fakulta, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovensko,
E-mailová adresa: `robert.sasik@uniza.sk`, `https://svf.uniza.sk/kgd/`

Ing. Dáša Smrčková, Katedra geodézie, Stavebná fakulta, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovensko,
E-mailová adresa: `dasa.bacova@uniza.sk`

Ing. Jakub Chromčák, PhD., Katedra geodézie, Stavebná fakulta, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovensko,
E-mailová adresa: `jakub.chromcak@uniza.sk`

doc. Ing. Jana Ižovltová, Dr., Katedra geodézie, Stavebná fakulta, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovensko,
E-mailová adresa: `jana.izvoltovak@uniza.sk`

12. ročník Žilina 2. – 4. 7. 2024
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSS Conf

OSSConf 2024: 77–82

SUDOKU S PŘEKRYVY: AHOJ, SVĚTE!

PAVEL STŘÍŽ (CZ)

Abstrakt. Autor stručně představuje proces generování sudoku s překryvy. V pozadí používá program Sugen, který byl pro tyto účely upraven na úrovni jazyka C. Na základním příkladu je proces představen krok za krokem. V závěru autor zmiňuje omezení tohoto algoritmu. V principu můžeme říci, že jakékoliv sudoku s překryvy, které obsahuje cyklus, tímto způsobem nelze generovat, je potřeba jiný přístup, a to přes rekurzivní funkci.

Klíčová slova. C, Sugen, sudoku.

MULTI-SUDOKU: HELLO, WORLD!

Abstract. The article briefly introduces a process of generating a multi-sudoku. Behind the scene, the author uses Sugen program which was modified at a C level for this specific task. A “Hello, World!” multi-sudoku is presented step-by-step. In the conclusion, the author mentiones its limits, esp. which multi-sudoku cannot be generated by this algorithm. It can be stated that any multi-sudoku with cycle would be a problem, this type of multi-sudoku needs a different approach, a recursion function.

Keywords. C, Sugen, sudoku.

1. Nápad

Řeším kombinatorické úlohy, především ranking-unranking problem, tedy zjistit počet možností, vybrat si jednu z nich, zrekonstruovat kombinatorickou situaci, a naopak, z jisté kombinatorické situace spočítat její pořadové číslo. To mě přivedlo i ke klasickému sudoku, kde jsem řešení nezahlédl, ale první experimenty ukazují, že to bude náročné, ale možné. O tom snad jindy.

Při rešerši kolem sudoku jsem narazil na verze s překryvy (anglicky overlapping sudoku, multi-sudoku). Pravděpodobně nejznámější je Samurai sudoku. Ale existuje jich celá plejáda.

2. Program sugen

Chtěl jsem si něco takového zkusit naprogramovat. Při rešerši jsem narazil na program Sugen (zkráceno ze sudoku generator) od Daniela Beera, `https://dlbeer.co.nz/articles/sudoku.html`.

Ten umí vygenerovat mustr, zadání sudoku ze zadaného mustru i získat obtížnější variantu řešeného sudoku. Neumí však generovat více sudoku s překryvy.

Poněvadž se učím C, tak jsem začal zdrojový kód zkoumat. Zkusit si vygenerovat sudoku s překryvy od nuly mě tehdy ani nenapadlo.

3. Hello, World!

Našim úkolem je připravit obdobu tohoto sudoku s překryvy. Překrývá se celý blok, to lze považovat za standard.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1																
2		4		5					2							
3		8														
4					8		7	4								
5				7	4			2		8						
6		2			7		9			6						
7		1		9			2	7								
8				1	9		4									9
9										1			8	2	7	
10			8					6		4				8	3	
11									6			2	3			
12								4				9				1
13											1	8			4	
14									9	6				4	5	
15									1	3	7			9		
16								2								6
17																

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1																
2		4	7	5	1	9	6	8	2	3						
3		8	9	6	2	4	3	5	1	7						
4		3	1	2	8	5	7	4	6	9						
5		6	3	7	4	1	5	2	9	8						
6		2	5	8	7	3	9	1	4	6						
7		1	4	9	6	8	2	7	3	5						
8		5	6	1	9	7	4	3	8	2	5	7	4	1	6	9
9		7	2	4	3	6	8	9	5	1	3	6	8	2	7	4
10		9	8	3	5	2	1	6	7	4	9	1	2	8	3	5
11								1	6	5	4	2	3	7	9	8
12								4	3	8	6	9	7	5	2	1
13								7	2	9	1	8	5	6	4	3
14								8	9	6	2	3	1	4	5	7
15								5	1	3	7	4	6	9	8	2
16								2	4	7	8	5	9	3	1	6
17																

4. Tři injekce od pana doktora (či sestřičky)

Jakmile jsem princip programu pochopil, použil jsem načtení externích souborů před zavoláním klíčové funkce. Připadal jsem si jako správný hacker.

1. injekce. Číslo sudoku slouží jako počáteční hodnota PRNG, funkce `srandom`. Pro případ pozdního nového generování jen jednoho sudoku. Nebylo třeba. Pomocný soubor obsahuje jednu hodnotu.

2. injekce. Pro vytvoření řešení, funkce `choose_grid` jsem zasahoval do pole `grid`. Pomocný soubor má dva sloupce, číslo pole (0–80) a hodnota (1–9). Vzniká soubor s řešením.

3. injekce. Pro vytvoření zadání, funkce `harden_puzzle` jsem zasahoval do pole `puzzle`. Pomocný soubor má dva sloupce, číslo pole (0–80) a jeho hodnotu (0 – musí zůstat prázdné, či 1–9). Plus se načte soubor s řešením. Výstupem je soubor se zadáním.

Poznámka. Dá se tak řešit i chtěná obtížnost, nebo program umožňuje si zavolat parametr `-t` z příkazového řádku. Toho jsem využil.

5. Dílčí kroky

Překryv je pro první sudoku 9. blok (pravý dolní roh), pro druhé sudoku blok 1. (levý horní roh). To když má člověk pořád na mysli, sledovat proces generování není pak tak náročné, hlavně u složitějších struktur.

Schématicky bychom generování mohli znázornit takto.

Upravený sugen: 1. sudoku, řešení:

```
_________                                   475196823
_________                                   896243517
_________                                   312857469
_________                                   637415298
_________   -->   bez injekce   -->         258739146
_________                                   149682735
_________                                   561974382
_________                                   724368951
_________                                   983521674
```

Upravený sugen: 1. sudoku, zadání:

```
475196823                                   4_5____2_
896243517                                   8________
312857469                                   ___8_74__
637415298                                   __74__2_8
258739146   -->   bez injekce   -->         2__7_9__6
149682735                                   1_9__27__
561974382                                   __19_4___
724368951                                   ________1
983521674                                   _8____6_4
```

Upravený sugen: 2. sudoku, řešení:

```
_________             382______             382574169
_________             951______             951368274
_________             674______             674912835
_________             _________             165423798
_________   --->      _________   --->      438697521
_________             _________             729185643
_________             _________             896231457
_________             _________             513746982
_________             _________             247859316
```

Upravený sugen: 2. sudoku, zadání:

```
382574169          000______          ________9
951368274          001______          __1__827_
674912835          604______          6_4___83_
165423798          _________          _6__23___
438697521   --->   _________   --->   4___9___1
729185643          _________          ___18__4_
896231457          _________          _96___45_
513746982          _________          _137__9__
247859316          _________          2_______6
```

Z příkazového řádku jsem si postupně dvakrát volal:

```
$ ./sugen gen-grid >reseni.txt
$ ./sugen harden -t 100 <reseni.txt >zadani.txt
```

Pro první sudoku byly dva pomocné soubory prázdné, pro druhé sudoku s naznačenými hodnotami u injekcí.

Poslední úkol je už čistě typografický, výstupní soubory si vhodně vysázet.

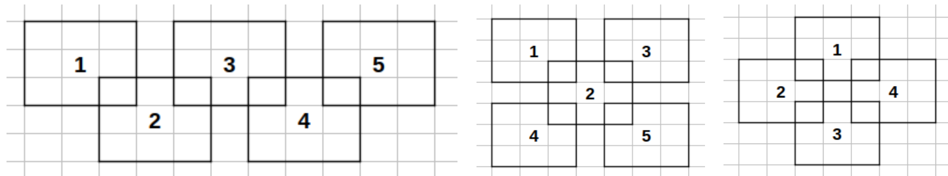
6. Kdy to nelze použít

Je zde omezení. Jakmile bychom měli komplikovanější strukturu překryvů (sudoku vztahy jakoby ob jedno sudoku), dříve či později se dostaneme do konfliktu, kdy musíme sudoku generovat znovu, či samozřejmě lépe, využít rekurze. O tom více na přednášce.

Pokud však najdeme pořadí sudoku, abychom jedno negenerovali ze vztahů více nezávislých sudoku, tato metoda nám dostačuje. Termínem z teorie grafů, do každého sudoku může jít maximálně jedna šipka, ze sudoku ven libovolný počet. Šipky určují pořadí generování.

Pro ilustraci dva náčrtky. Na levém obrázku můžeme generovat sudoku např. v těchto pořadích: 1–2–3–4–5, 5–4–3–2–1 nebo třeba 3–2–1–4–5. Do problémů se dostáváme např. při 1–2–3–5–4 či 5–1–2–3–4, ale také při pokusu generovat sudoku jakoby po řádcích, tedy 1–3–5–2–4.

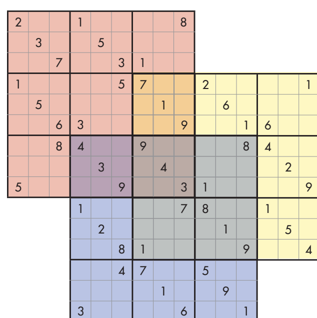
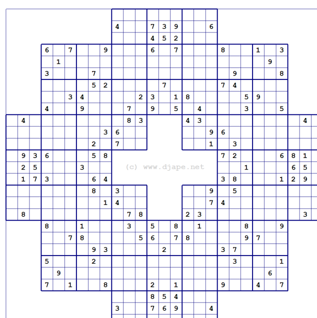
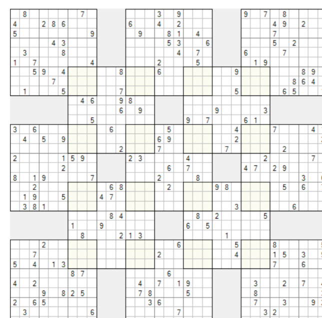
Na středním obrázku můžeme využít: 1–2–3–4–5 či 2–1–3–4–5, ale dostaneme se do problémů při 1–3–2–4–5 či 1–3–4–5–2.



Tato pomůcka sledu generování sudoku nám nepomůže u multi-sudoku, které tvoří uzavřený cyklus, kdy každé sudoku má víc nezávislých překryvů, viz pravý obrázek.

Na takovou situaci se už musí jít přes rekurzivní funkci. Či mnou nedoporučený způsob vyřazování nevhodného sudoku a opakování. V takovém případě se ale může stát, že řešení ani nalézt nelze a musí se jít v procesu generování o celé sudoku nazpět. Chce to jiný přístup, o tom víc na konferenci.

Tento typ jsem neprogramoval, neb jsem to zatím nepotřeboval, ale kdyby na to došlo, zde je tzv. Venn sudoku (*Taking Sudoku Seriously*, str. 122), které je ideálním typem na testy. Plus další dva rozsáhlejší cyklické typy přebrané z internetu z různých zdrojů (Sudoku Gattai a Sudoku Sumo).

7. Rešerše zdrojů: Logická hra sudoku

Články

Rád bych upozornil na tři články:

- Arnab Kumar Maji et al.: An Exhaustive Study on Different Sudoku Solving Techniques. *International Journal of Computer Science Issues*, Vol. 11, Issue 2, No. 1, March 2014. ISSN 1694-0814, eISSN 1694-0784. `https://www.ijcsi.org/papers/IJCSI-11-2-1-247-253.pdf`
- Mária Ercsey-Ravasz, Zoltán Toroczkai: *The Chaos Within Sudoku,* arXiv: 1208.0370v1, August 1, 2012. `https://arxiv.org/pdf/1208.0370.pdf`
- Gary McGuire, Bastian Tugemann, Gilles Civario: *There is no 16-Clue Sudoku: Solving the Sudoku Minimum Number of Clues Problem via Hitting Set Enumeration,* arXiv 1201.0749v2, August 31, 2013. `https://arxiv.org/pdf/1201.0749.pdf`

Knihy

Za pozornost stojí tyto čtyři knihy.

- *Taking Sudoku Seriously,* a,
- *Geometric Magic Squares.*
- Wei-Meng Lee: *Programing Sudoku,* Apress, USA, 2006. ISBN 978-1-59059-662-3. Zdrojové kódy jsou na `https://github.com/Apress/programming-sudoku`.

- Giulio Zambon: *Sudoku Programming with C,* Apress, USA, 2015. ISBN 978-1-4842-0996-7. Zdrojové kódy jsou dostupné na `https://github.com/apress/sudoku-programming-w-c`.

Efektivní algoritmus

Efektivní algoritmus vyvinul D. E. Knuth známý jako DLX, Dancing Links, Algorithm X. Knuth, Donald E. (2000). Dancing links. *Millennial Perspectives in Computer Science.* P159. 187. arXiv 0011047v1, November 15, 2000. `https://arxiv.org/pdf/cs/0011047.pdf`

Volně dostupné programy

V Linuxu jsou k dispozici balíky:

- `sgt-puzzles`, příkaz `sgt-solo`.
- `qqwing`, příkaz ten stejný.
  Umí vypsat kroky u řešení: `qqwing --generate 1 --instructions`.
- `gnome-sudoku`, příkaz stejný.
- `nbsdgames`, příkaz `nbsudoku`. Tip: `nbsudoku -s 7`.
- `sudoku`, příkaz stejný.
- `ksudoku`, příkaz stejný. Umí řadu typů, včetně 3D.
- `fltk1.3-games`, příkaz `flsudoku`.

Program Sugen byl zmíněn v tomto článku, vlastní program viz `https://dlbeer.co.nz/articles/sudoku.html`.

Na GitHubu má uživatel KyleGough svůj program `sudoku`, který řeší sudoku pomocí logických, resp. hráčských metod, viz `https://github.com/KyleGough/sudoku`.

Další zajímavé zdroje

Počty různých sudoku, viz `http://www.afjarvis.org.uk/sudoku/`.

Řešení různých variant sudoku, viz kanál na YouTube `https://www.youtube.com/@CrackingTheCryptic`.

Guinnessova kniha rekordů, `https://www.guinnessworldrecords.com/`. Zajímavé a aktivní jsou tři: Largest multi-sudoku puzzle (280 sudoku), Most people playing sudoku simultaneously (3824) a Largest sudoku published (jedná se o sudoku 100 na 100).

Kontaktní adresa

Ing. Pavel Stříž, Ph.D., U Škol 940, Bučovice, okres Vyškov, 685 01, Česká republika,
E-mailová adresa: `pavel@striz.cz`

12. ročník
Žilina 2. – 4. 7. 2024
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
osscconf.soit.sk
OSS Conf
OSSConf 2024: 83–88

PROBLÉM N SUDOKU CIFER

PAVEL STŘÍŽ (CZ)

Abstrakt. V článku je představeno rozšíření klasického programátorského problému známého jako problém N dam. Autor ukáže program pro Python3 s rekurzí a zpětným vyhledáváním všech možností plus výsledky svých experimentů a možnosti dalšího bádání.

Klíčová slova. Sudoku, rekurze, zpětné vyhledávání, problém N dam.

N SUDOKU DIGITS PUZZLE

Abstract. The article introduces an extension to a classical programming problem known as the N queens puzzle. The author shows the program for Python3 with recursion and backtracking of all possibilities. He also mentions results of his experiments and open problems.

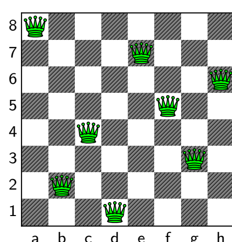
Keywords. Sudoku, recursion, backtracking, N queens puzzle.

1. Problém N dam

Problém, který v článku představím, je rozšířením problému N dam. Je to oblíbený problém soutěžních programátorů. Pamatuji si, že když jsem byl prcek, tak program pro QBasic jsem uměl nazpaměť, jak jsem si to neuměl dobře představit a naprogramovat, jak se sluší a patří. Po letech se tomu musím zasmát, kdy už tehdy jsem se nebál žádné výzvy, byť byla evidentně nad mé síly.

Nejběžnější varianta je pro $N = 8$ a definice je prostá. Úkolem je rozmístit na šachovnici právě osm dam, ale tak, aby se vzájemně nenapadaly: horizontálně (řádek), vertikálně (sloupec) ani v žádné diagonální linii. Zde je ukázka jednoho z 92 řešení. Vedle typograficky a šachově hezkého schématu zobrazuji textovou verzi se znakem Q pro dámu a hvězdičku pro prázdné políčko, a pak se znakem 1 reprezentující dámu a 0 reprezentující prázdné pole. Je to z důvodu, aby se čtenář rychleji dostal do nitra programu, kde se snažím o zobecnění. Z figur bílého přejdu na barvu zelenou, aby bylo zřejmé, že se nejedná už o čistě šachový problém.

K vysázení šachovnice je užit balíček chessboard, kde jsem poprvé zahlédl vrstvení znaků k dosažení efektu barevné šachovnice a figur, nyní běžný přístup přes TikZ a balíčky jako jsou tikzducks, tikzpingus, tikzmarmots či tikzlings.

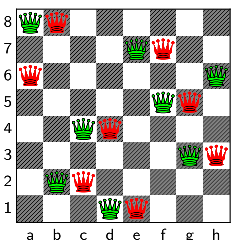



```
Q*******    10000000
****Q***    00001000
*******Q    00000001
*****Q**    00000100
**Q*****    00100000
******Q*    00000010
*Q******    01000000
***Q****    00010000
```

První důležitý postřeh pro budoucí program je, že na volné políčko může útočit více dam. Proto při návratu zpět z rekurze (odebrání dámy) nesmíme zakázané políčko uvolnit jen s ohledem na odebíranou dámu. Proto neužiji typ True/False, ale počet, kolik dam na dané políčko útočí. Jen tehdy, když je počet nula, tak na dané políčko lze dámu umístit.

2. Problém N sudoku cifer

Poněvadž se zabývám sudoku a jejími variantami, napadlo mě, jestli by nebylo možné přidat další várku dam, řekněme dámy černého hráče. Ty mezi sebou také nesmí útočit, ale dámy černé s bílými si nezavazí. V terminologii sudoku cifry 1 a 2. Vedle barvy zelené užiji barvu červenou, opět pro zdůraznění, že se dostáváme nad rámec hry v šachy.




```
Qq******    12000000
****Qq**    00001200
q******Q    20000001
*****Qq*    00000120
**Qq****    00120000
******Qq    00000012
*Qq*****    01200000
***Qq***    00012000
```

Otázka zní, zda-li je možné umístit všech 9 sudoku cifer. Pokud nikoliv, jaký je nejvyšší možný počet cifer. Z naší ukázky můžeme říci, že pro $N = 9$ určitě půjde umístit cifry 1–2, protože jsme našli alespoň jedno řešení. V dalších částech textu upouštím od sazby přes balíček `chessboard`, víc typů dam než dvě se tam neužívá. Dá se to obejít různou barvou písma, víc o tom na přednášce.

3. Program

Naše úvaha by mohla být tato. Umístíme první cifru rekurzí. Vezmeme postupně výsledky a rekurzí se do volných polí pokusíme umístit další cifru. Abychom však mohli program zobecnit, uděláme si vektor `pole` s devíti 1, devíti 2 atd. Plus si přidáme informaci, na kterém řádku se cifra může pohybovat. Zabrané pole cifrou bude v mapě `reseni`, tam už se nesmí umístit nic dalšího. Dílčí zákazy pro danou cifru budou v mapě `zakazy` s indexem cifry. Jinými slovy

`reseni` je globální zákaz pro všechny třídy, proměnná `zakazy` pak lokální zákaz pro danou třídu.

Druhý postřeh je, že zákazy nemusíme dávat do řádku (tam se pohybuje cifra), ale ani do řádků nad, zákazy dáváme od zkoumané cifry níž.

Program zobecníme tím, že si N volíme. U kombinatorických úloh si častokrát volím $L = N - 1$ (pomůcka L pro less z angličtiny), abych se z indexování od jedné dostal na indexování od nuly. Zde podobně užijeme $M = N + 1$ (pomůcka M pro more z angličtiny), abychom si hlídali horní mez v Pythonu, funkce `range`, ta horní mez nebere včetně.

Ukázka je nastavena na prvním řádku pro $N = 8$ a první dvě cifry (dámy).

```
N=8; M=N+1; C=2+1 # číslo+1, nebo N-2+1, nebo M (N+1)
reseni=[[0 for k in range(N)] for l in range(N)]
citac=0; zakazy=[0]*M
for k in range(1,M):
  zakazy[k]=[[0 for k in range(N)] for l in range(N)]

pole=[]
for k in range(1,C): # M, nebo N-2 + 1
  for tradek in range(N):
    pole.append([k,tradek])
lenpole=len(pole)

def printreseni():
  for tradek in range(N):
    for tsloupec in range(N):
      print(reseni[tradek][tsloupec],end="")
    print()

def mal(pozice):
  global citac
  if pozice==lenpole:
    citac=citac+1; print("Řešení č.",citac)
    printreseni(); print()
    return
  zkoumane=pole[pozice]
  cifra=zkoumane[0]; radek=zkoumane[1]
  for sloupec in range(N):
      if reseni[radek][sloupec]==0: # pole je volné
        if zakazy[cifra][radek][sloupec]==0: # není zde cifrový zákaz
          reseni[radek][sloupec]=cifra # sem dám cifru
          # sloupec, řádky pod pozicí
          for R in range(radek+1,N):
            zakazy[cifra][R][sloupec]+=1
          # doleva dolů
          usloupec=sloupec+1; uradek=N-radek; u=min(usloupec,uradek)
          for kroku in range(1,u):
            Nradek=radek+kroku; Nsloupec=sloupec-kroku
            zakazy[cifra][Nradek][Nsloupec]+=1
          # doprava dolů
          vsloupec=N-sloupec; vradek=N-radek; v=min(vsloupec,vradek)
          for krokv in range(1,v):
```

```
      Mradek=radek+krokv; Msloupec=sloupec+krokv
      zakazy[cifra][Mradek][Msloupec]+=1

    mal(pozice+1) # rekurze

    # vrátit vše zpět
    reseni[radek][sloupec]=0
    for R in range(radek+1,N):
      zakazy[cifra][R][sloupec]-=1
    for kroku in range(1,u):
      Nradek=radek+kroku; Nsloupec=sloupec-kroku
      zakazy[cifra][Nradek][Nsloupec]-=1
    for krokv in range(1,v):
      Mradek=radek+krokv; Msloupec=sloupec+krokv
      zakazy[cifra][Mradek][Msloupec]-=1

mal(0) # Spuštění rekurze
```

4. Získané výsledky

Pro $N = 1$ je jedno řešení, 1.

Pro $N = 2$ a $N = 3$ nemáme řešení.

Podobně pro $N = 4$, nejvyšší možný počet cifer je 2. Řešení jsou dvě pro jednu i dvě cifry.

```
0120   0210
2001   1002
1002   2001
0210   0120
```

Pro $N = 5$ dostáváme 240 řešení, první a poslední jsou následující. Zároveň vidíme, že jsou vertikálně překlopená jako pro $N = 4$.

```
12345   54321
45123   32154
23451   15432
51234   43215
34512   21543
```

Dle cifer: 10, 40, 120, 240 a 240.

Pro $N = 6$ nemáme řešení, nejvyšší počet cifer jsou 4. Dostáváme 24 řešení, zde jsou první dvě.

```
012340   012430
304102   403102
420031   320041
130024   140023
201403   201304
043210   034210
```

Dle cifer: 4, 12, 24, 24, 0 a 0.

Pro $N = 7$ dostáváme 20160 řešení. Zde jsou první dvě.

```
1234567   1234576
6712345   7612345
```

```
4567123    4576123
2345671    2345761
7123456    6123457
5671234    5761234
3456712    3457612
```

Dle počtu cifer: 40, 496, 2400, 5280, 11520, 20160 a 20160.

Pro $N = 8$ nemáme řešení. Nejvyšší počet cifer je 6 a získáme 12960 řešení. V takovém případě dostáváme první dvě řešení tato.

```
12345600    12346500
45003126    46003125
06254031    05264031
30160254    30150264
60530412    50630412
03412065    03412056
21006543    21005643
54621300    64521300
```

Můžeme shrnout, že pro jednu cifru máme 92 řešení (problém osmi dam), pro dvě 3252, pro tři 37224, pro čtyři 109080, pro pět 44160, pro šest již zmíněných 12960, pro sedm a osm 0.

Pro $N = 9$. Tohle je jádro mých snah: získat podklady pro skutečné sudoku. Zjistil jsem následující. Můžeme shrnout, že pro jednu cifru máme 352 řešení (problém devíti dam), pro dvě 48328, pro tři 2315208, pro čtyři 33563424, pro pět až devět se mi to zatím nedopočítalo.

Pozn. Ve funkci `printreseni` jsem si upravil řádek na:

```
print(hex(reseni[tradek][tsloupec])[2:].upper(),end="")
```

abych místo 10 měl A, místo 11 B apod.

Pro $N = 10$. Ve výpočetním čase mne daném jsem nebyl schopen projít všechny možnosti a potvrdit, že to řešení nemá. Pro osm cifer by první dvě řešení byla tato. Zjistit počet řešení jsem se již nesnažil.

```
1203456078    1203456087
0512637840    0512638740
4730810625    4830710625
5687002134    5678002134
8304275061    7304285061
7065184302    8065174302
2840360517    2740360518
0156723480    0156823470
6071548203    6081547203
3428001756    3427001856
```

Pro $N = 11$ dostáváme řešení neuvěřitelně rychle. První řešení je tzv. žebřík, druhé také s prohozenými A a B. Zjistit počet řešení bylo nad mé výpočetní síly.

```
123456789AB    123456789BA
AB123456789    BA123456789
89AB1234567    89BA1234567
6789AB12345    6789BA12345
456789AB123    456789BA123
```

```
23456789AB1    23456789BA1
B123456789A    A123456789B
9AB12345678    9BA12345678
789AB123456    789BA123456
56789AB1234    56789BA1234
3456789AB12    3456789BA12
```

Zkusil jsem ještě $N = 12$ (počet cifer deset) a $N = 13$ (najít alespoň jedno řešení), ale bylo to již moc silné kafé.

5. Domněnka a otevřené problémy

Lze si ze získaných výsledků stanovit pracovní hypotézu:

- Pro $N \in \mathbb{N}$ lichá bez trojky: existuje řešení pro počet cifer 1 až N.
- Pro $N \in \mathbb{N}$ sudá bez dvojky: existuje řešení pro počet cifer 1 až $N - 2$.

Něco takového potvrdit či vyvrátit je nad mé matematické a výpočetní možnosti. Zdá se však, dle charakteru úlohy, že pro libovolné $N \geq 4$ je počet řešení stejný pro N a $N - 1$.

Bylo by zajímavé zkusit heuristiku místo rekurze, např. jsou-li pro větší N dostupná řešení typu žebřík.

Pro $N = 9$, má-li řešení, by bylo zajímavé zjistit, jestli je to platné řešení pro sudoku. Tedy po získání řešení vybrat jen ta, která navíc splňují podmínku sudoku bloků 3×3.

Podobně jako u problému N dam by bylo zajímavé se zabývat počtem řešení se zahrnutím symetrie (rotace, překlápění).

Z pohledu programování: Navyšoval jsem počet cifer přes proměnnou `C` po jedné a zapisoval si počet řešení. Bylo by efektivnější zasáhnout do rekurze, dát `C=M` a vypadla by tabulka s počty řešeními pro všechny možnosti 1 až N.

Za typografii by bylo příjemné si jednotlivé třídy vysázet samostatně automatizovaně, aby neútočení bylo lépe vidět, např. u prvního řešení pro $N = 5$:

```
1****   *2***   **3**   ***4*   ****5
**1**   ***2*   ****3   4****   *5***
****1   2****   *3***   **4**   ***5*
*1***   **2**   ***3*   ****4   5****
***1*   ****2   3****   *4***   **5**
```

To je nad rámec tohoto článku a mého cíle představit tento problém. Žhavé numerické novinky od mých výpočetních strojů zmíním v Žilině. Na viděnou!

Kontaktní adresa

Ing. Pavel Stříž, Ph.D., U Škol 940, Bučovice, okres Vyškov, 685 01, Česká republika,
E-mailová adresa: `pavel@striz.cz`

12. ročník Žilina 2. – 4. 7. 2024
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
osssconf.soit.sk
OSS Conf
OSSConf 2024: 89–98

VÍTEJTE VE SVĚTĚ ANIMACÍ!

PAVEL STŘÍŽ (CZ)

Abstrakt. Článek představuje základní možnosti animování grafiky ve světě TEXu.

Klíčová slova. Metapost, animate, PSTricks, Asymptote, TikZ, dvisvgm, svganimation, media4svg.

WELCOME TO THE WORLD OF ANIMATIONS!

Abstract. The article introduces basic options of animating graphics in the TEX world.

Keywords. Metapost, animate, PSTricks, Asymptote, TikZ, dvisvgm, svganimation, media4svg.

1. O animacích

V dobách dřívějších se na webové stránky často dávaly animované gify. Tyto dny se složením jednotlivých obrázků a jejich extrakcí z gifů pomáhá ImageMagick či odnož GraphicsMagick.

Druhý oblíbený formát je Flash. Firma Adobe však končí s podporou programu Flash Player v prosinci 2020. Tedy například tyto animace `https://melusine.eu.org/syracuse/metapost/animations/` se nám hned tak v budoucnu nepodaří otevřít. Na Linuxu lze na přehrání užít program `gnash`.

```
$ sudo apt install gnash
```

Co se týče zařazení animace do pdf, tak jednu z možností přes JavaScript zmínili J. Holeček a P. Sojka v článku Animations in pdfTEX-generated PDF ve sborníku *TEX, XML, and Digital Typography*, Springer, str. 179–191, 2004. O rok později to zmiňuje i J. Gilg v článku PDF-Animationen v časopisu *Die TEXnische Komödie,* Vol. 17, No. 4, str. 30–37, 2005. Podpůrný balíček interactiveplot vzniká roku 2014 a vzniká balík AcroTEX, některé části jsou zadarmo, některé nabízené za poplatek.

Obecně se může animace uložit jako (audio)videostopa. K tomu nám slouží především balíky `ffmpeg` a ve starších linuxových distribucích `avconv`.

```
$ sudo apt install ffmpeg
```

Ve světě open source software existuje nespočet nástrojů na přehrání videa, např. `mpv`, `vlc` a pro Raspberry Pi optimalizovaný `omxplayer`.

```
$ sudo apt install mpv vlc-bin
```

Zařazení audiovideo stop do pdf nabízel TEXový balíček movie15 a nyní jeho nástupce balíček media9. Vedle toho umožňují zařadit soubory s Flash animacemi a 3D objekty (PRC, U3D).

```
$ texdoc media9 movie15
```

2. animate v2020-04-25

Vrcholem v TEXovém světě je balíček animate, který umožňuje zařadit animace vznikající vrstvením obrázků na sebe, jejich případné časování a výběr kreslených částí, parametr timeline (to je výhodné u rozsáhlých obrázků skrz velikost výsledné animace) a nově pomáhá s generováním animovaných svg. Zkusme si prvně získat animace ve čtyřech základních nástrojích dostupných v TEXLive u ukázek mimo TEXLive.

2.1. METAPOST v2.0

Dokumentaci získáme přes

```
$ texdoc metapost metafun-p
```

Jedna z nejstarších galerií je od Vincent Zoonekynd z roku 1999 http://zoonek.free.fr/LaTeX/Metapost/metapost.html a archiv na https://www.ctan.org/tex-archive/info/metapost/examples

Jednoduchou ukázku vzniku animace přes sérii obrázku nalezneme na adrese https://adityam.github.io/context-blog/post/metapost-animation

Pokročilé animace hledejme na adresách http://www-math.univ-poitiers.fr/~phan/animations.html, https://melusine.eu.org/syracuse/metapost/animations.

Díky knihovně luamplib umíme psát kód METAPOSTu přímo v TEXovém dokumentu, zájemce nechť nahlédne na tuto ukázku: https://melusine.eu.org/syracuse/luatex/luamplibAnimate

Při problémech s písmy na úrovni METAPOSTu se doporučuje užít v preambuli prologues:=3. Ukážeme si animaci vykreslení celého odstavce ze zmíněné galerie. Jen se mi nepodařilo ji vygenerovat přes balíček luamplib přímo z TEXového dokumentu, podezřívám násobnou inicializaci proměnných. https://melusine.eu.org/syracuse/metapost/animations/mehats

Soubor 010.mp vypadá takto:

```
filenametemplate "%j-%3c.mps";
verbatimtex%&latex
\documentclass{article}
\usepackage{lmodern} \usepackage[utf8]{inputenc} \usepackage[T1]{fontenc}
\begin{document}
etex;
picture tex_pct, glp_pct; numeric glp_num, pth_num[]; path glp_pth[][];
```

```
tex_pct:=btex{\begin{minipage}{\textwidth}\begin{center}
Ukázka animace spojených sil\\balíčků METAPOST a animate!
\end{center}\end{minipage}}etex;
glp_pct:=nullpicture;
string fnt_str, txt_str, sub_str; numeric txt_wd; glp_num:=0;
for tkn within tex_pct:
  if textual tkn:
    fnt_str:=fontpart tkn; txt_str:=textpart tkn; txt_wd:=0;
    for glp_idx=0 upto (length txt_str-1):
      sub_str:=substring (glp_idx, glp_idx+1) of txt_str;
      pth_num[glp_num]:=0;
      for sub_tkn within glyph ASCII sub_str of fnt_str
          scaled (fontsize fnt_str/1000) xscaled xxpart tkn
          yscaled yypart tkn shifted (txt_wd+xpart tkn, ypart tkn):
        glp_pth[glp_num][pth_num[glp_num]]:=pathpart sub_tkn;
        addto glp_pct doublepath glp_pth[glp_num][pth_num[glp_num]];
        pth_num[glp_num]:=pth_num[glp_num]+1;
      endfor
      glp_num:=glp_num+1; txt_wd:=txt_wd+
        (xxpart tkn)*xpart urcorner (sub_str infont fnt_str);
    endfor
  fi
endfor
numeric bg_wd, bg_hg; picture bg_pct; bg_wd:=1280; bg_hg:=300; bg_pct:=nullpicture;
addto bg_pct contour origin--(bg_wd, 0)--(bg_wd, bg_hg)--(0, bg_hg)--cycle;
numeric fg_wd, fg_hg; transform fit_trn;
fg_wd:=xpart(urcorner glp_pct-llcorner glp_pct); fg_hg:=ypart(urcorner glp_pct-llcorner
     glp_pct);
fit_trn:=identity shifted -.5[llcorner glp_pct, urcorner glp_pct]
  scaled .9min(bg_wd/fg_wd, bg_hg/fg_hg) shifted +.5[llcorner bg_pct, urcorner bg_pct];
color bg_clr, fg_clr; pen fg_pen; numeric dot_scl; bg_clr:=white;
fg_clr:=black; fg_pen:=pencircle scaled 2; dot_scl:=4;
numeric duration, fps, f_num; duration:=10; fps:=25; f_num:=fps*duration;
for idx=0 upto (f_num/2-1):
  beginfig(idx)
  draw bg_pct withcolor bg_clr; drawoptions (withcolor fg_clr);
  for i=0 upto glp_num-1:
    for j=0 upto pth_num[i]-1:
      path pth; numeric tim; pth:=glp_pth[i][j] transformed fit_trn;
      tim:=arctime 2(arclength pth)/f_num*idx of pth;
      draw subpath (0, tim) of pth withpen fg_pen;
      draw point (tim) of pth withpen fg_pen scaled dot_scl;
    endfor
  endfor
  drawoptions ();
  endfig;
endfor
end.
```

Pomocný soubor je 010-metapost.tex:

```
\documentclass{article}
\usepackage{animate}
\usepackage{graphicx}
\begin{document}
\animategraphics[width=0.75\textwidth, controls=all, poster=last]{10}{010-}{000}{124}
\end{document}
```

Spouštíme:

```
$ mpost 010.mp
$ lualatex 010-metapost.tex; lualatex 010-metapost.tex
```

Ukázka animace spojených sil
balíčků METAPOST a animate!



2.2. PStricks v2.97 a nespočet jeho balíčků

Galerie najdeme na stránkách programu: http://tug.org/PSTricks/main.cgi?file=packages

Na animace se častokrát používá pomocný balíček multido, ukázky ze světa PSTricks najdeme přímo v balíčku animate. Z galerií vypíchněme:

https://tug.org/PSTricks/main.cgi?file=Animation/gif/gif,

https://tug.org/PSTricks/main.cgi?file=Animation/basics,

https://melusine.eu.org/syracuse/pstricks/pst-solides3d/animations.

```
$ texdoc multido animate
```

Zvláštní kategorii tvoří server s blogy http://pstricks.blogspot.com. Narazil jsem na celou řadu zajímavých balíčků, např. xint. Na serveru je představena celá řada vznikajích a pracovních balíčků. Zmíním vybrané.

Dle vzoru https://geargenerator.com vzniká balíček pst-gears, v poslední verzi v0.6. Verze pro 2D je ke stažení na: http://manuel.luque.free.fr/pst-gears-2020/pst-gear-2020-v0.6.zip, nebo https://drive.google.com/drive/folders/1zyXX3w525m99YPM4wkSd3acJRbcCVs4o. Verze pro 3D, pst-gearsiiid, ve verzi v3, je dostupná na: http://manuel.luque.free.fr/gearsIIID/pst-gearsIIID-v3.zip, https://drive.google.com/open?id=1sSIVv2rqbFHhCkyX_VvZ5oKLIMXxdrv2.

Zaujal mě i balíček pst-crayon, v3.1, ze kterého si přebereme ukázku. https://drive.google.com/open?id=0Bw5_RBu0n8-qbkhrVGN1REVRUGs.

Soubor `020.tex` vypadá takto:

```
\documentclass[pstricks]{standalone}
\usepackage{pst-plot,pst-3d,pst-gears,pst-node}
\usepackage[nomessages]{fp} \makeatletter
\define@key[psset]{}{theta1}{\def\psk@thetaA{#1}}
\define@key[psset]{}{theta2}{\def\psk@thetaB{#1}}
\psset{theta1=-90,theta2=90}
\def\psElasticFixedTwoWheels{\pst@object{psElasticFixedTwoWheels}}
\def\psElasticFixedTwoWheels@i{{\begin@SpecialObj
\FPset{\ZA}{\psk@ZA}\FPset{\ZB}{\psk@ZB} \FPset{\module}{\psk@m}
\FPeval{\RA}{\ZA*\module/2}\FPeval{\RB}{\ZB*\module/2} \FPeval{\OB}{\RA+\RB}
\FPeval{\RAp}{(\RA*2-2.5*0.2)/2} \FPeval{\RBp}{(\RB*2-2.5*0.2)/2}
\FPset{\OMEGAA}{-1} \FPeval{\OMEGAB}{(-\OMEGAA)*\ZA/\ZB}
\FPset{\ANGLE}{\psk@wheelrotation} \FPeval{\ANGLErad}{\ANGLE*\FPpi/180}
\FPeval{\nombrePoints}{trunc(2*\ANGLE+5,0)}
\FPeval{\thetaA}{(\psk@thetaA)*\FPpi/180} \FPeval{\thetaB}{(\psk@thetaB)*\FPpi/180}
\FPeval{\xA}{0.9*\RAp*cos(\thetaA+\OMEGAA*\ANGLErad)}
\FPeval{\yA}{sin(\thetaA+\OMEGAA*\ANGLErad)*\RAp*0.9}
\FPeval{\xB}{cos(\thetaB+\OMEGAB*\ANGLErad)*\RBp*0.9}
\FPeval{\yB}{sin(\thetaB+\OMEGAB*\ANGLErad)*\RBp*0.9+\OB}
\FPeval{\xM}{(\xA+\xB)/2} \FPeval{\yM}{(\yA+\yB)/2}
\ThreeDput[normal=0 0 1](0,0,0){\psgrid[subgriddiv=0,gridlabels=0pt]
\rput(0.05,-0.05){\pstgears[circles=false, polarangle=90, fillstyle=solid, color1=black,
     color2=black]}
\pstgears[circles=false,polarangle=90,fillstyle=solid]
\parametricplot[linecolor=red, plotpoints=\nombrePoints, algebraic,
     linewidth=0.1]{0}{\ANGLErad}{
  (\RAp*0.9*cos(\thetaA+\OMEGAA*t)+\RBp*0.9*cos(\thetaB+\OMEGAB*t))/2|
  (\RAp*0.9*sin(\thetaA+\OMEGAA*t)+\RBp*0.9*sin(\thetaB+\OMEGAB*t)+\OB)/2}
\psline{->}(0,0)(0,1) \psline{->}(0,0)(1,0)
\pscircle[linestyle=dotted](0,0){\RA} \pscircle[linestyle=dotted](0,\OB){\RB}}
\ThreeDput[normal=0 1 0](\xA,\yA,0) {\psline[linewidth=0.1]{-*}(0,0)(0,1) \pnode(0,1){P1}}
\ThreeDput[normal=0 1 0](\xB,\yB,0) {\psline[linewidth=0.1]{-*}(0,0)(0,1)\pnode(0,1){P2}}
\ThreeDput[normal=0 1 0](\xM,\yM,0) {\pnode(0,1){P3} \pnode(0,0){P4}}
\psline[linecolor=blue](P1)(P2) \psline[linecolor=red]{->}(P3)(P4)
     \psdot[linecolor=blue](P3)
\end@SpecialObj}\ignorespaces} \makeatother
\begin{document}
\multido{\i=0+45}{17}{\begin{pspicture}(-5,-5)(5,6)
\psElasticFixedTwoWheels[Z1=35, Z2=10, m=0.15, viewpoint=-1 -2 2, arrowinset=0,
     arrowsize=0.2, wheelrotation=\i, linewidth=0.025, color1=yellow, color2=green]
\end{pspicture}}
\end{document}
```

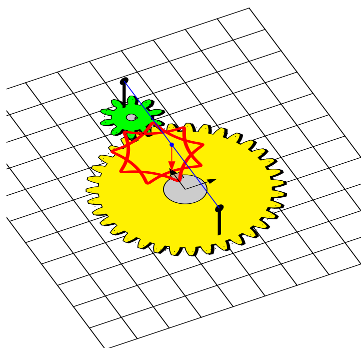
Pomocný soubor je `020-pstricks.tex`:

```
\documentclass{standalone}
\usepackage{animate}
\usepackage{graphicx}
```

```
\begin{document}
\animategraphics[width=0.5\textwidth,controls=all,poster=last]{1}{020}{}{}
\end{document}
```

Spouštíme:

```
$ latex 020.tex
$ dvips 020.dvi
$ ps2pdf 020.ps
$ lualatex 020-pstricks.tex; lualatex 020-pstricks.tex
```





2.3. Asymptote v2.65

V galerii programu `https://asymptote.sourceforge.io/` je blok animací: `https://asymptote.sourceforge.io/gallery/animations`.

Zaujala mě galerie P. Ivaldiho na `http://asy.marris.fr/asymptote/` s animacemi: `http://asy.marris.fr/asymptote/animations/index.html`. Zde je ještě jedna galerie `http://www.piprime.fr/asymptote/` s animacemi: `http://www.piprime.fr/developpeur/asymptote/animation-asy_asy`.

Vybral jsem následují ukázku. `http://www.piprime.fr/1208/animation_asymptote-fig0090/`.

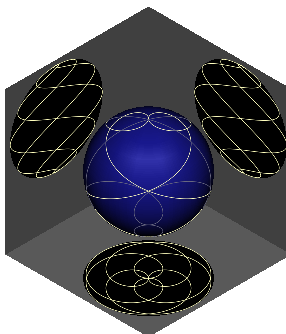
Dočasně jsem skrz generování gifů vyhodil bezpečnostní pravidla:

```
$ cd /etc/ImageMagick-6/
$ sudo mv policy.xml policy-old.xml
```

Soubor `030.asy` vypadá takto:

```
size(0,10cm); import graph3; import animation; import solids;
currentlight.background=black; settings.render=0;
animation A; A.global=false; int nbpts=500; real q=2/5; real pas=5*2*pi/nbpts;
int angle=4; real R=0.5; pen p=rgb(0.1,0.1,0.58); triple center=(1,1,1);
transform3 T=rotate(angle,center,center+X+0.25*Y+0.3*Z);
real x(real t){return center.x+R*cos(q*t)*cos(t);}
real y(real t){return center.y+R*cos(q*t)*sin(t);}
```

```
real z(real t){return center.z+R*sin(q*t);}
currentprojection=orthographic(1,1,1); currentlight=(0,center.y-0.5,2*(center.z+R));
triple U=(center.x+1.1*R,0,0), V=(0,center.y+1.1*R,0);
path3 xy=plane(U,V,(0,0,0)); path3 xz=rotate(90,X)*xy;
path3 yz=rotate(-90,Y)*xy; triple[] P; path3 curve; real t=-pi;
for (int i=0;i<nbpts;++i){t+=pas;triple M=(x(t),y(t),z(t));P.push(M);curve=curve..M;}
     curve=curve..cycle;
draw(surface(xy),grey); draw(surface(xz),grey); draw(surface(yz),grey);
triple xyc=(center.x,center.y,0); path3 cle=shift(xyc)*scale3(R)*unitcircle3;
surface scle=surface(cle); draw(scle, black);
draw(rotate(90,X)*scle, black); draw(rotate(-90,Y)*scle, black);
draw(surface(sphere(center,R)),p); triple vcam=1e5*currentprojection.camera-center;
for (int phi=0; phi<360; phi+=angle) {bool[] back,front; save();
  for (int i=0; i<nbpts; ++i) {P[i]=T*P[i];bool test=dot(P[i]-center,vcam)>0;
     front.push(test);}
  curve=T*curve; draw(segment(P,front,operator ..), paleyellow);
  draw(segment(P,!front,operator ..),0.5*(paleyellow+p));
  draw((planeproject(xy)*curve)^^ (planeproject(xz)*curve)^^
    (planeproject(yz)*curve), paleyellow); A.add(); restore();}
A.movie(options="-density 350 -resample 96 -quality 100 -depth 8 -strip");
```





Pomocný soubor je `030-asymptote.tex`:

```
\documentclass{article}
\usepackage{animate}
\usepackage{graphicx}
\begin{document}
\animategraphics[width=0.5\textwidth,controls=all,poster=last]{1}{_030+}{0}{89}
\end{document}
```

Spouštíme:

```
$ asy -vk 030.asy
$ for soubor in `find -iname _030\*.eps`; do
> core=${soubor%.eps}
> echo $soubor; ps2pdf $soubor
> pdfcrop --hires $core; mv $core-crop.pdf $core.pdf
> done
$ lualatex 030-asymptote.tex; lualatex 030-asymptote.tex
```

2.4. TikZ v3.1.5b

TikZ si získal nemalou oblibu. Má rozsáhlou dokumentaci.

```
$ texdoc tikz
```

Největší galerie TEXample, se skládá z příspěvků mnoha uživatelů. http://www.texample.net/tikz/examples/tag/animations/. Zde jsem vybral některé animace z oblasti matematiky a statistiky.

http://www.texample.net/tikz/examples/animated-set-intersection/,

http://www.texample.net/tikz/examples/animated-definite-integral/,

http://w...example.net/tikz/examples/convolution-of-two-functions/,

http://www.texample.net/tikz/examples/animated-distributions/,

http://w...et/tikz/examples/sine-and-cosine-functions-animation/.

TikZ samotný však není vhodný nástroj na 3D grafy, neumí skrývat neviděné části, není na to primárně stavěný. S tím do velké míry pomáhá balíček pgfplots, aktuálně ve verzi v1.17, a pomocný balíček pgfplotstable, v1.17.

```
$ texdoc pgfplots pgfplotstable
```

Za zmínku stojí galerie, sourozenec TEXample, server http://pgfplots.net. Spojil jsem tyto dvě ukázky, 3D graf a animaci. http://pgfplots.net/tikz/examples/bivariate-normal-distribution a https://tex.stackexchange.com/questions/266125/animate-a-pgfplots-3d-plot.

Soubor 040.tex vypadá takto:

```
\documentclass{article}
\pagestyle{empty}
\usepackage{pgfplots}
\pgfplotsset{width=8cm, height=6cm, compat=1.17}
\pgfplotsset{colormap={whitered}{color(0cm)=(white); color(1cm)=(orange!75!red)}}
\begin{document}
\foreach \malAngle in {40,50,...,400}{%
\newpage
\begin{tikzpicture}[
  declare function = {mu1=1;}, declare function = {mu2=2;},
  declare function = {sigma1=0.5;}, declare function = {sigma2=1;},
  declare function = {normal(\m,\s)=1/(2*\s*sqrt(pi))*exp(-(x-\m)^2/(2*\s^2));},
  declare function = {bivar(\ma,\sa,\mb,\sb) = 1/(2*pi*\sa*\sb) * exp(-((x-\ma)^2/\sa^2 +
    (y-\mb)^2/\sb^2))/2;}]
  \draw (-1.5cm,-1cm) rectangle (9.5cm,5cm);
  \begin{axis}[colormap name=whitered, view={\malAngle}{65}, enlargelimits=false,
     grid=major, domain=-1:4, y domain=-1:4, samples=26, xlabel=$x_1$, ylabel=$x_2$,
     zlabel={$P$}, colorbar, colorbar style={at={(1.25,0.4)}, anchor=east, height=2cm,
     title ={$P(x_1,x_2)$}}]
    \addplot3 [surf] {bivar(mu1,sigma1,mu2,sigma2)};
    \addplot3 [domain=-1:4,samples=31, samples y=0, thick, smooth]
      (x,4,{normal(mu1,sigma1)});
```

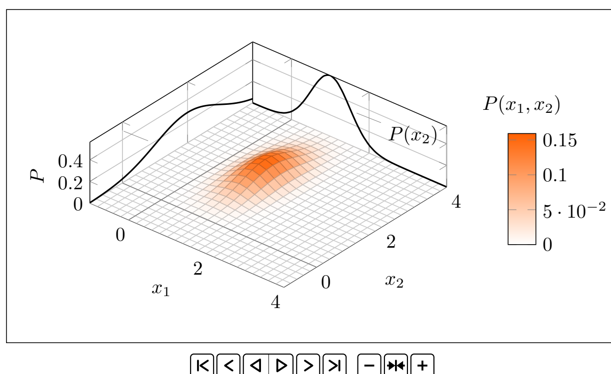
```
    \addplot3 [domain=-1:4,samples=31, samples y=0, thick, smooth]
      (-1,x,{normal(mu2,sigma2)});
    \draw [black!50] (axis cs:-1,0,0) -- (axis cs:4,0,0);
    \draw [black!50] (axis cs:0,-1,0) -- (axis cs:0,4,0);
    \node at (axis cs:-1,1,0.18) [pin=165:$P(x_1)$] {};
    \node at (axis cs:1.5,4,0.32) [pin=-15:$P(x_2)$] {};
  \end{axis}
\end{tikzpicture}}
\end{document}
```

Pomocný soubor je `040-tikz.tex`:

```
\documentclass{article}
\usepackage{animate}
\usepackage{graphicx}
\begin{document}
\animategraphics[width=0.75\textwidth,controls=all,poster=last]{10}{040}{}{}
\end{document}
```

Spouštíme:

```
$ lualatex 040.tex; lualatex 040.tex
$ pdfcrop --hires 040.pdf
$ mv 040-crop.pdf 040.pdf
$ lualatex 040-tikz.tex; lualatex 040-tikz.tex
```



3. Okular v20.04-1: zobrazení animace

Vznik animace je jedna věc, jak je zobrazit v pdf je věc druhá. Velký problém ve svobodném světě softwaru je, jak takové pdf s animacemi zobrazit. Adobe zrušilo podporu Readeru pro Linux v dubnu 2013 u verze 9.5.5 pro 32bitové počítače. FoxIt Reader sice animace vzniklé z balíčku animate umí zobrazit, ale také jen mimo Linux. Prakticky stejně je na tom prohlížeč PDF-XChange Viewer.

U odlehčených prohlížečů pdf (XpdfReader, MuPDF, Okular, Evince) jsme neměli šanci. Uživatel Linuxu to musí obcházet: míchání 32 a 64bitových aplikací, Wine, přes virtuální stroj či zobrazením pdf na stroji bez Linuxu.

Poměrně velký mezník znamená Google Summer of Code 2019, kdy João Netto rozšiřuje Okular a animace vzniklé přes animate lze spustit. https://community.kde.org/GSoC/2019/StatusReports/Jo%C3%A3oNetto

Ani po velkém úsilí, se mi nepodařilo ze zdrojových kódů

```
$ git clone https://cgit.kde.org/okular.git
```

dostat takovou verzi, která by si s tím poradila (Ubuntu 18.04, Ubuntu 20.04, Debian 10). Nepodařilo se mi to ani přes

```
$ sudo snap install --edge okular
```

Můj nejlepší odhad je, že je nevhodná verze knihovny programu Poppler. Ovšem nahlédneme-li na zařazení nové verze 20.04.1 u distribucí https://okular.kde.org/download.php máme vyhráno. Nastartujeme-li Ubuntu 20.10, Arch či Gentoo, vše běží jako po másle přímo z linuxového repozitáře.

Opatrně! Je zde však řešení i pro starší distribuce. Na Xubuntu 18.04 jsem v `/etc/apt/sources.list` přidal
`deb http://cz.archive.ubuntu.com/ubuntu/ groovy main universe`
a ostatní vstupní body jsem si zakomentoval. Pak jsem si vzal na pomoc nástroj `aptitude` a po určité době hledání a řešení konfliktních balíčků se mi podařilo nástroj nainstalovat. Sledujte však pozorně, co chce nástroj odinstalovat, aby to nebyla většina linuxové distribuce.

```
$ sudo apt update
$ sudo aptitude install okular
```

Několik postřehů. Animace nejedou přes prezentační režim, ale dá se ze Settings skrýt Toolbar, Navigation Panel a Page Bar a přejít do celoobrazovkového režimu přes Ctrl+Shift+F.

Po nakliknutí Show Forms úvodní mávající smajlík balíčku animate se rozběhne až po zarolování na jinou stranu a zpět. Naopak při Hide Forms zůstává stále aktivní.

Vylepšený Okular nabízí zobrazení pdf, ps, djvu, tiff, chm i formátu epub. Může se hodit i na zobrazení textových souborů, například datových, aux a log souborů při běžné práci. U svých experimentů jej používám i na zobrazení dvi souborů.

Na zobrazení swf či 3D objektů v prohlížeči pdf si ve svobodném softwarovém světě ještě počkáme, doporučuji prozatím Adobe Reader.

Kontaktní adresa

Ing. Pavel Stříž, Ph.D., U Škol 940, Bučovice, okres Vyškov, 685 01, Česká republika,
E-mailová adresa: pavel@striz.cz

12. ročník Žilina 2. – 4. 7. 2024
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSS Conf
OSSConf 2024: 99–108

PF2020?! ANEB KDYŽ NESTAČÍ ANI R, ANI TEX

PAVEL STŘÍŽ (CZ)

Abstrakt. Článek zmiňuje úvahy a kroky vedoucí k vysázení loga PF2020! Logo je vysázené z šestiúhelníkových nálepek, více o kolekcích v GitHub repozitářích hex-stickers (Rstudio) a BiocStickers (Bioconductor). Hlavní zdroj inspirace bylo logo z konference useR! 2018, které vytvořil Mitchell O'Hara-Wild za použití jeho R skriptu hexwall.

Klíčová slova. R, ImageMagick, hexwall, raster, gglogo, magick, tidyverse, ggplot2, sf, TEX, Lua, Bash, GraphicsMagick, png, pacman.

PF2020?! OR WHEN NEITHER R NOR TEX ARE SUFFICIENT

Abstract. The article describes a thinking process and a problem-solving approach of creating a PF2020! logo. It's typeset of hexagon stickers, see Rstudio's hex-stickers and Bioconductor's BiocStickers repositories in GitHub. The main inspiration came from the useR! 2018 logo which was created by Mitchell O'Hara-Wild using his hexwall R script.

Keywords. R, ImageMagick, hexwall, raster, gglogo, magick, tidyverse, ggplot2, sf, TEX, Lua, Bash, GraphicsMagick, png, pacman.

1. Problém typu word cloud

Word cloud / wordle / tag cloud se běžně řeší tak, že slova či obrázky se převedou do rastrových obrázků a následně se zkoumají přesahy na úrovni pixelů[1]. Dlouhodobě se zabýváme vektorovým řešením tohoto problému na úrovni TEXu. To lze zrealizovat např. přes METAPOST a příkaz `bisect`, ale to je vše „na dlouhé lokte“. Proto nás zaujalo logo z konference userR! 2018, které má svou mřížku z šestiúhelníků, ale nálepky (angl. stickers) jsou sázeny jen uvnitř polygonů, v jejich případě mapy AustrálieNávod lze nalézt na blogu autora, viz webová stránka `https://www.mitchelloharawild.com/blog/user-2018-feature-wall/`.

2. První dojmy

Zkusili jsme v tomto duchu připravit PF2020! v R, co nejrychleji… Ale! Shrneme-li problémy: je potřeba doinstalovat knihovny v R a k tomu je potřeba mít nástroje a knihovny. Užili jsme Linux a omlouváme se uživatelům Microsoft Windows, že jsme testy nezkoušeli pod tímto operačním systémem. Instalovali jsme jednotlivé knihovny jednu po druhé a sledovali v R chybové zprávy. Tímto

[1] `https://www.jasondavies.com/wordcloud/about/`

způsobem jsme se vrátili pod Linux a doinstalovali vždy to, co bylo potřeba. A opětovně instalovali konkrétní knihovnu v R. Jednalo se postupně o knihovny

`libmagick++-dev`, `librsvg2-dev`, `libssl-dev`, `libogdi3.2-dev`, `libcurl4-openssl-dev`, `gdal-bin` a `libwebp-dev`.

Je však možné, že jsme některé nástroje již měli nainstalované, proto není výčet úplný. Funkce `hexwall` je závislá na knihovně magick, který nám pro velký počet souborů přestane fungovat. V našem případě kolem tisícího souboru. Nehledě na časovou náročnost práce s obřím rastrovým obrázkem v pozadí.

Další problém byl, že se nám za půlden testů zaplnil pracovní adresář `/tmp/` o 500 GB. Hledejme proto jiné nástroje.

Poslední problém byl, že jsme nechtěli mapu (geografická data), ale texty, případně užít obecný (černobílý) rastrový obrázek. Na tohle jsme se zaměřili.

3. Zdárné kroky s knihovnou gglogo

Při hledání převodu znaků do polygonů jsme objevili tuto knihovnu. Pracovali jsme v adresáři, kam jsme si uložili skript hexwall:

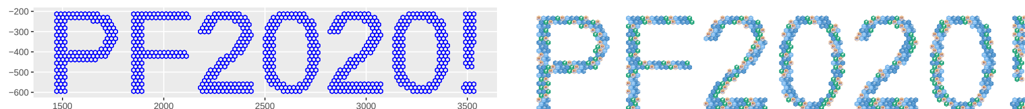
```
$ git clone https://github.com/mitchelloharawild/hexwall.git; \
> cd hexwall
```

Ačkoliv jsme zápasili s převody mezi datovými typy, zde je použitelný výsledek. Mezi pokusy jsme na vyčištění používali příkaz `rm(list=ls())`.

```
library(gglogo); library(raster); library(sf)
letter <- letterToPolygon("PF2020!", fontfamily="Helvetica", dim=c(5000,800))
Sr1 = Polygon(cbind(letter$x,letter$y)); Srs1=Polygons(list(Sr1),"s1")
SpP = SpatialPolygons(list(Srs1), 1:1)
hex_points <- SpP %>% spsample(type = "hexagonal", cellsize = 20)
hex_points@coords

library(ggplot2); library(tidyverse); as_tibble(hex_points@coords)
aus_hex <- HexPoints2SpatialPolygons(hex_points, dx = 20)
#pdf("nahled.pdf")
ggplot() + geom_sf(data=st_as_sf(aus_hex), colour="blue", fill=NA)
#dev.off()

source("hexwall.R")
mojkovo <- hexwall("samplehex", sticker_width=20, coords=hex_points@coords,
    sort_mode="filename")
plot(mojkovo)
#image_write(mojkovo,"pf2020-pres-hexwall.png")
```



V této ukázce se nám nelíbilo, že je tam málo nálepek (5 souborů), nemůžeme užít zúženou mezeru za PF i netradičně před vykřičník, abychom poškádlili typografický svět, a chtěli jsme si zkusit výběr bez vracení s vracením (vysvětlíme). Pokusme se v další části článku tyto úkoly vyřešit.

4. První krok s knihovnou raster

Jistým vývojovým mezistupněm se nám stala „zakázka“, chceme-li experiment, pro organizátory konference OSSConf v Žilině, `http://ossconf.soit.sk/`. Vzali jsme logo roku 2019, zvětšili, v GIMPu zasáhli do roku, drobně jsme roztáhli cifry a vyčistili vyhlazování typické u obrázků na internetu (Colors→Threshold) plus úprava pixelů „zde, tu a támhle“. Tím jsme si zajistili obrázek přesně se čtyřmi barvami a bílým pozadím (v případě nutnosti průhledným).

Zkusili jsme načíst rastrový obrázek a do vzniklých polygonů vkreslit šestiúhelníky. Abychom se procvičili v R, zkusili jsme cyklus `for` přes barvy. Přes `data[]` jsme zjistili, že červená barva je v intenzitách 66, 55, 153 a 77. Příslušné RGB hodnoty jsme vyčetli v GIMPu. Zkušenější uživatelé R by jistě přišli na to, jak unikátní RGB hodnoty získat přímo v R a na tom postavit cyklus. Ručně to u 4 barev šlo, kdyby jich bylo víc, bylo by potřeba celý postup zautomatizovat.

Nepodařilo se nám hexa hodnotu barev vytáhnout z `data.frame` po spojení proměnných `hodnoty` a `barvy`. Zůstalo nám to jako otevřený problém.

Zde je výsledek našich snah.

```
library(raster); library(ggplot2); library(sf)
data<-raster("tux/ossconf-2020-upravene.png"); hodnoty<-c(66,55,153,77)

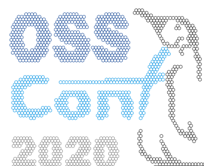

barvy<-c("#426baa","#37a9e9","#999999","#4d4d4d")
mojkovo<-ggplot()
for (volim in 1:length(hodnoty)) {
  polygony <- rasterToPolygons(data, dissolve=TRUE,
    fun=function(x){x==hodnoty[volim]})
  hexiky <- polygony %>% spsample(type = "hexagonal", cellsize = 8)
  nahled <- HexPoints2SpatialPolygons(hexiky, dx = 8)
  mojkovo <- mojkovo + geom_sf(data=st_as_sf(nahled), colour=barvy[volim],
    fill=NA)
  }
mojkovo <- mojkovo + theme_void(); mojkovo
#pdf("2020-logo-ossconf.pdf"); mojkovo; dev.off()
```



Ručně ⟶



R ⟶



5. Druhý krok s knihovnou raster

Připravili jsme si obecný rastrový obrázek. Začali jsme v TEXu ve složce sazba sázet soubor pf2020.tex:

```
\documentclass{article}
\pagestyle{empty}
\usepackage{graphicx}
\begin{document}
\centering\bf\sffamily
\resizebox{9.5mm}{!}{PF}\par2020!\par ČStS%
\end{document}
```

Získali jsme postupně pdf a poté tif soubor v linuxovém prostředí:

```
$ lualatex pf2020.tex; \
> pdfcrop --hires --margins 5 pf2020.pdf; \
> gm convert -density 300 -monochrome pf2020-crop.pdf pf2020-crop.tif
```

V R jsme užili tento skript:

```
library(raster)
library(tidyverse)
library(sf)
data <- raster("sazba/pf2020-crop.tif")
polygony <- rasterToPolygons(data, dissolve=TRUE, fun=function(x){x>0})
hexiky <- polygony %>% spsample(type = "hexagonal", cellsize = 1.5)
as_tibble(hexiky)
nahled <- HexPoints2SpatialPolygons(hexiky, dx = 1.5)
ggplot() + geom_sf(data=st_as_sf(nahled), colour="brown", fill=NA)
```

Zajistili jsem si tak, že můžeme pracovat s libovolným černobílým rastrovým obrázkem a libovolným rozlišením. V této chvíli jsme však neužili skript hexwall, ale vyexportovali jsme si nalezené souřadnice středů šestiúhelníků.

```
write.table(hexiky@coords, "hexagony.csv", sep=",", col.names=FALSE)
```

První řádky souboru hexagony.csv vypadaly takto:

```
"1",62.7959799161181,21.6272033299488
"2",64.2959799161181,21.6272033299488
"3",65.7959799161181,21.6272033299488
```

6. Sesypání šestiúhelníkových nálepek

Objevili jsme dva velké repozitáře s nálepkami, stáhli jsem si je přes:

```
$ git clone https://github.com/rstudio/hex-stickers.git; \
> git clone https://github.com/Bioconductor/BiocStickers.git
```

Nakopírovali jsme si png do nové složky `pfhex`, u hex-stickers to bylo rychlé. U BioStickers jsme použili pomocný skript v Lua, který nám naparsoval soubor `README.md` a vytáhl si png soubory nálepek z podadresářů:

```
soubor=io.open("README.md")
obsah=soubor:read("*all")
soubor:close()
kam=io.open("spust.sh","w")
unicode.utf8.gsub(obsah, "<img src=\"([^\"]-%.png)\"", function(s)
  print(s)
  kam:write("cp "..s.." ../hexwall-master/pfhex\n")
  end)
kam:close()
```

Soubor jsme spustili přes `texlua` dostupný v TEXLive a vzniklý dávkový soubor přes `sh`.

Posledním úkolem bylo připravit si podklady. Připravili jsme si složku `pfhex-output` a spustili následující (šikovný administrátor si snadno převede do skriptu):

```
$ cd pfhex; \
> for file in `find -type f -iname \*.png -printf "%f\n"`; do \
>   echo $file; \
>   gm convert $file -transparent white ../pfhex-output/$file; \
> done
```

Tím jsme zajistili, že jsou soubory průhledné, nezlobí tam barevný profil ICC (ImageMagick) a můžeme operativně zasáhnout do velikosti obrázků přes parametr `resize`.

7. Luujeme

Nyní máme stavební kameny a opustíme R. V TEXu je práce s datovými soubory možná, my použijeme Lua skript a v TEXu budeme jen sázet výsledek. Je to obdoba generování HTML přes PHP či JavaScript. Kvůli čitelnosti však nemícháme Lua skript uvnitř TEXu, což je možné, ale oddělíme jej.

Lua jako skriptovací jazyk se stal neocenitelným pomocníkem v TEXovém světě. Příchod LATEXu3, nemluvě o ConTEXtu, sice umí mnohé, ale pro „obyčejné“ programátory je Lua jen jiná forma C++, Javy, JavaScriptu, Pythonu či Perlu. Ukažme si prvně střípky programování v Lua.

7.1. Výběr s vracením

Kdybychom chtěli vybrat 13 čísel od 1 do 52, můžeme to učinit takto:

```
for k=1,13 do
  print(math.random(1,52))
end -- končí cyklus for
```

Takový soubor bychom spustili přes `texlua` z TEXLive. Nabízí se možnost programu `lua`, např. z balíku `lua5.2`, případně `lua5.3` z balíku `lua5.3`.

7.2. Výběr bez vracení

Vytvoříme si prázdné pole a vyplníme jej hodnotami 1 až 52. Postupně volíme pořadí z pole a vypisujeme hodnotu na dané pozici. Tuto hodnotu pak z pole odebereme.

```
hodnoty={}
for x=1,52 do table.insert(hodnoty,x) end
for k=1,13 do
  vyber=math.random(1,#hodnoty)
  print(hodnoty[vyber])
  table.remove(hodnoty,vyber)
end -- končí cyklus for
```

7.3. Výběr bez vracení s vracením

Taková vánoční drobnost. U novoročenky jsme chtěli výběr bez vracení, ale nalezených šestiúhelníků jsme měli víc než nálepek. Místo nějaké formy stratifikovaného výběru jsme z pole hodnoty odebírali a jakmile bylo pole prázdné, vyplnili jsme si jej všemi dostupnými nálepkami znovu. Tím jsme zajistili, že jsou výběry náhodné, ale že se žádná nálepka neopakuje výrazně vícekrát než jiné. Základem Lua je práce s tabulkami, při jejich kopírování přebíráme jednotlivé položky (angl. deep copy), prosté „rovná se“ by nám nepomohlo.

Pojďme na školní ukázku. Z 52 čísel jich vybereme 117. Znak `#` nám zjistí aktuální velikost pole.

```
hodnoty={}; vsechny={}
for x=1,52 do
  table.insert(hodnoty,x)
  table.insert(vsechny,x)
end -- končí cyklus for
for k=1,117 do
  vyber=math.random(1,#hodnoty)
  io.write(hodnoty[vyber].." ") -- místo print
  table.remove(hodnoty,vyber)
  if #hodnoty==0 then -- deep copy
    for k,v in pairs(vsechny) do hodnoty[k]=v end
  end -- končí podmínka if
end -- končí cyklus for
```

Náš pokus by mohl dopadnout takto: 34 13 35 46 7 12 48 43 [...] 16 5 28.

Pozorný čtenář jistě brzy zjistí, že 39 hodnot se opakuje přesně dvakrát, 13 hodnot přesně třikrát. Obdoba by byla, když rozdáváme balíček karet, a jakmile jsme všechny karty rozdali, použijeme další balíček karet. Kdybychom rozdávali po 13

kartách, rozdali jsme karty 9 hráčům, spotřebovali bychom dva celé a jednu čtvrtinu balíčku žolíkových karet bez žolíků.

8. R + Lua + TEX

Bokem si uložíme z adresáře `pfhex-output` seznam nálepek:

```
$ ls *.png >../soubory-pfhex.txt
```

V R jsme provedli výpočty a získali jsme soubor `hexagony.csv`. Nyní si přes Lua skript tyto dva soubory načteme. Náš cíl je vygenerovat Ti*k*Zový zdrojový kód, který vysázíme. Navíc jsme si v Lua skriptu nastavili jednoduché měřítko. Výsledek našich snah by mohl vypadat takto:

```
math.randomseed(1)
soubory=io.open("soubory-pfhex.txt") -- seznam nálepek
obsah=soubory:read("*all")
soubory:close()
pngs={}
pngsfull={}

unicode.utf8.gsub(obsah, "([^\n]-)\n", function(s)
  table.insert(pngs,s) -- pracovní tabulka s nálepkami
  table.insert(pngsfull,s) -- neměnná tabulka všech nálepek
  end)
soubor=io.open("hexagony.csv") -- seznam nalezených souřadnic šestiúhelníků
obsah=soubor:read("*all")
soubor:close()
kam=io.open("zdrojak.tex","w") -- TikZový zdrojový soubor
kam:write("\\begin{tikzpicture}[remember picture, overlay]\n")
unicode.utf8.gsub(obsah, ",([^,]-),([^\n]-)\n", function(s,t)
  tos=s/1; tot=t/1 -- jednoduchá škála, je-li nutná
  pickup=math.random(1,#pngs) -- výběr bez vracení
  kam:write("  \\node[m] at ("..tos.."pt,"..tot.."pt)
    {\\includegraphics[width=\\maldimen]{"..pngs[pickup].."}};\n")
  table.remove(pngs,pickup) -- odebere vybrané logo
  if #pngs==0 then -- vrátit všechny nálepky, deep copy
    for k,v in pairs(pngsfull) do pngs[k] = v end -- for
  end -- if
end) -- unicode.utf8.gsub
kam:write("\\end{tikzpicture}\n")
kam:close()
```

Postupně se generuje soubor `zdrojak.tex`, první řádky vypadají takto:

```
\begin{tikzpicture}[remember picture, overlay]
  \node[m] at (62.795979916118pt,21.627203329949pt)
    {\includegraphics[width=\maldimen]{glue.png}};
```

```
  \node[m] at (64.295979916118pt,21.627203329949pt)
    {\includegraphics[width=\maldimen]{scmap.png}};
```

Náš poslední úkol je načíst si tento Ti*k*Zový kód a vysázet novoročenku, soubor `pf.tex`. To provedeme v TEXu. My jej měli ve složce `sazba`, aby se nám pomocné TEXové soubory nemíchaly s ostatními.

```
\documentclass[landscape]{article}
\pagestyle{empty}
\usepackage{tikz}
\graphicspath{ {../pfhex-output/} }
\newdimen\maldimen \maldimen=1.5pt % cellsize / škála v Lua souboru
\tikzset{inner sep=0pt, outer sep=0pt,
  m/.style={xshift=-100pt, yshift=-100pt, draw=none} }

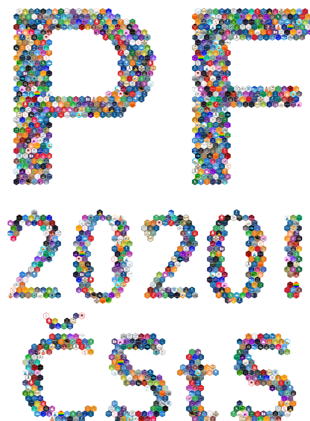
\begin{document}
\input ../zdrojak.tex
\end{document}
```

Nezbývá než si vše vysázet:

```
$ lualatex pf.tex; \
> lualatex pf.tex; \
> pdfcrop --hires --margins 0 pf.pdf; \
> gm convert -density 1800 pf-crop.pdf pf-crop.png
```

První dva řádky zajistí vysázení a absolutní umístění na straně. Třetí řádek nám ořeže ochrannou bílou zónu. Šikovný TEXista brzy zjistí, že by se to dalo zjednodušit na jeden běh TEXu bez nástroje `pdfcrop`. Necháváme otevřené pro badatele. Poslední řádek nám vygeneruje rastrový náhled.

Nezbývá než se pokochat vánočním dárkem, a doufat, že soubory pdf a png nebudou moc velké, nebude se to mezi svátky dlouho vykreslovat a do Nového roku se novoročenka celá zobrazí…



9. Závěrečné tipy: ještě jedno ohlédnutí za R

S určitým časovým odstupem si odpovídáme na otevřené otázky u R.

Při instalaci knihovny rgdal to chce novější verzi R než nabízí standardní linuxový repozitář (Xubuntu 18.04). Vyřešili jsme to takto.

V /etc/apt/sources.list jsme přidali:

```
deb https://cloud.r-project.org/bin/linux/ubuntu bionic-cran35/
```

Následovala instalace R:

```
$ sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys
    E298A3A825C0D65DFD57CBB651716619E084DAB9
$ sudo apt update
$ sudo apt upgrade
$ sudo apt install r-cran-base
```

V případě problémů se závislostmi užíváme místo známých programů apt či apt-get program aptitude. Hodilo se nám to při míchání 32 a 64bitových aplikací, resp. programů z různých linuxových distribucí.

V R následuje doinstalování knihoven, které použijeme. Ideální řešení je instalovat jednu knihovnu za druhou a sledovat případné chybové zprávy.

```
install.packages(c("png","spex","raster","rgdal","sf","pacman"))
```

9.1. Zjištění barvy konkrétního pixelu

Poněvadž jsme zápasili s datovými typy, zkusili jsme si získat barvu konkrétního pixelu a dostat jeho RGB složky, ve stylu GIMPu, ale už bez GIMPu.

Využili jsme knihovnu raster a funkci brick nebo stack, získali jsme složky RGB, převedli na šestnáctkové hodnoty a přidali znak #.

```
library(raster)
data<-brick("tux/ossconf-2020-upravene.png")  # nebo
#data<-stack("tux/ossconf-2020-upravene.png")
danaBarva<-paste("#", paste( as.hexmode( getValues(data,50,1)[50,] ), sep="",
    collapse=""), sep="")
danaBarva
```

Výstup je:

```
[1] "#426baa"
```

To už lze přímo použít pro parametr colour, např. colour=danaBarva.

9.2. Výpis všech barev v obrázku

Máme za sebou barvu jednoho pixelu, podívejme se, jak lze proces zautomatizovat a získat všechny jedinečné barvy v rastrovém RGB obrázku. Použijeme knihovnu png.

```
library(png)
mujpng<-readPNG("tux/ossconf-2020-upravene.png")
unique(as.raster(mujpng[,,1:3]))
```

Výstup je:

```
[1] "#FFFFFF" "#4D4D4D" "#426BAA" "#37A9E9" "#FEFEFE" "#FEFFFF" "#999999"
```

U našeho obrázku linuxového tučňáka Tuxe v logu konference OSSConf jsme zjistili, že máme dvě barvy (#426BAA, #37A9E9), dvě šedé (#4D4D4D, #999999), bílé pozadí (#FFFFFF) a na první pohled neviditelné, řekněme „parazitní", chceme-li přehlédnuté, téměř bílé pixely: šedé a barevné (#FEFEFE, #FEFFFF).

Nyní už lze barvy oddělovat (například chceme mít co barva to jeden polygon), filtrovat (nechceme např. bílé pozadí ani padesát odstínů šedi), měnit (nahradit jednu barvu za jinou), spojovat (např. spojit barvy textů, nebo vše spojit do jedné barvy) ap.

9.3. Pracovní zobrazení obrázku

Zkusíme si jednoduchý filtr: odstranit bílé a téměř bílé pixely a zobrazit si pracovní náhled obrázku. Zároveň pro načtení více R knihoven otestujeme knihovnu pacman. Zmíněné otevřené problémy v R jsou tímto uzavřené.

```
#library(raster); library(sf); library(spex); library(ggplot2)
library(pacman) # alternativní postup načtení více knihoven
pacman::p_load("raster","sf","spex","ggplot2")
data<-raster("tux/ossconf-2020-upravene.png")
data[data>200] <- NA # ořez bílých a téměř bílých pixelů
vysledek<-polygonize(data)
#plot(vysledek) # je to pomalé, ale použitelné
#pdf("pracovni-nahled.pdf") # uložení pro bulletinek
ggplot()+geom_sf(data=st_as_sf(vysledek))
#dev.off() # uzavření ukládání pdf
```



Kontaktní adresa

Ing. Pavel Stříž, Ph.D., U Škol 940, Bučovice, okres Vyškov, 685 01, Česká republika,
E-mailová adresa: pavel@striz.cz

12. ročník Žilina 2. – 4. 7. 2024 Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach osscconf.soit.sk OSS Conf
OSSConf 2024: 109–116

NOVINKY ZE SVĚTA R+KORONAVIRU

PAVEL STŘÍŽ (CZ)

Abstrakt. Článek je stručný popis myšlenek z doby koronaviru, kdy jsem pracoval s R. Pojďme trochu zavzpomínat na tu dobu.

Klíčová slova. R, CRAN, Bioconductor.

NEWS FROM THE WORLD OF R+CORONAVIRUS

Abstract. This article briefly describes my memories of the coronavirus period while working with R. Let us refresh the memories!

Keywords. R, CRAN, Bioconductor.

The R Foundation Retweeted: Peter Dalgaard. R 4.0.0 "Arbor Day" (source version) has been released.

24. 4. 2020 mi přistála na stole tato zpráva a za pár dní na to, 28. 4. 2020, došlo k aktualizaci Bioconductoru na verzi 3.11. Je Svátek práce, jdu ty nové `lazyLoad`, `lazyEval` a `lazyData` v R vyzkoušet a sdělit svůj nezávislý pohled.

1. R v4.0.0

Bez otálení jsem na svém Xubuntu 20.04 do `/etc/apt/sources.list` přidal:

```
deb https://cloud.r-project.org/bin/linux/ubuntu bionic-cran40/
```

Zakomentoval jsem starší pokusy a provedl preventivní kroky:

```
$ sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys
    E298A3A825C0D65DFD57CBB651716619E084DAB9
$ sudo apt update
$ sudo apt upgrade
```

Jádro aktualizace pak tvořil příkaz:

```
$ sudo apt install r-base r-base-dev
```

Rychlý test prokázal, že se podařilo a provedl jsem aktualizaci knihoven:

```
$ R --version
$ R
> update.packages(.libPaths())
```

2. COVID v19

The R Foundation Retweeted: R Consortium has started new GitHub repository to centralize collaboration and data sources – looking to develop COVID-19 tools and code – Come add your information and contribute to the community! `https://bddy.me/3aAXOmb`

Z toho samého dne zaujala mou pozornost ještě tato zpráva. Řekl jsem si, že bych měl nově nainstalované R hlouběji vyzkoušet.

Na úvodní stránce na mne vyskočily 4 projekty: Coronavirus Tracker, COVID-19 Propagation, COVID-19 Tracker Map a COVID-19 Projections.

Vezmu-li to od konce. U Projections na mne vyskočil GitHub. Po určitém bádání se mi podařilo otevřít rozcestník a webovou stránku pro Českou republiku `https://www.volzinnovation.com/covid-19_SARS-CoV-2_corona/reports/latest/Czechia.html`. Autorem je Raphael Volz.

Autorem Tracker Map je Jay Ulfelder. V RStudiu Cloud mne hned upozornili, že se jedná o dočasný projekt. Pod Shiny app na mne vedle analýz vyškočily pdf v záložce WHO Situation Reports. Zajímavý nápad.

Druhý projekt v pořadí je Propagation. Autorem je Juan Francisco Venegas Gutiérrez. Bohužel repozitář na GitHubu mi nešel otevřít, tak jsem to zahlásil (Issues). Mezi modely jsem Českou republiku nenašel.

3. Coronavirus Tracker v0.1.4

První projekt v pořadí od Johna Coeneho Coronavirus Tracker vyzývá ke spuštění R. Pustil jsem se do toho. RStudio cloud občas jel bez přístupových práv, požadavek na knihovny `shinyMobile` a `echarts4r` zněl zajímavě.

```
$ R
> install.packages("coronavirus")
```

Zkusil jsem z dokumentace první ukázku a ještě si vyžádal knihovnu `dplyr`. Proč si ji nenainstaloval sám?

```
> install.packages("dplyr")
> library(coronavirus)
> require(dplyr)
> coronavirus %>% filter(type=="confirmed") %>% group_by(Country.Region) %>%
    summarise(total=sum(cases)) %>% arrange(-total) %>% head(20)
# A tibble: 20 x 2
   Country.Region       total
   <chr>                <int>
 1 Mainland China       70446
 2 Others                 355
 3 Singapore               75
```

Tady jsem zbystřil. To jsou stará data v textové formě, nikoliv hezké mapy přes web s aktuálními daty. Ve slangové řeči: rtfm! To, co jsem právě nainstaloval,

je knihovna `https://github.com/RamiKrispin/coronavirus`, která má stejný název. Mezi Issues jsem autorovi Trackeru zahlásil, že jeho název je v konfliktu s existující knihovnou z února 2020.

Pokračoval jsem v experimentování, knihovnu jsem odinstaloval a podíval se na návod, `https://coronavirus.john-coene.com`.

```
> remove.packages("coronavirus")
> install.packages("remotes")
> remotes::install_github("Johncoene/coronavirus")
```

Během instalace mi naskočila tato neobvyklá zpráva:

```
- Use `usethis::browse_github_pat()` to create a Personal Access Token.
- Use `usethis::edit_r_environ()` and add the token as `GITHUB_PAT`.
```

Knihovna `usethis` se teprve instalovala. V každém případě zmínka o `Rate limit reset at: [...]`, kdy došlo k restartu za několik minut pro mne znamenalo chvíli počkat a instalaci zopakovat.

Před koncem instalace si R vyžádalo systémový balík `libpq-dev` ve starší verzi 10.3-1 (aktuální je 10.12-0). Udělal jsem hrubý krok, doporučuji čtenářům najít lepší řešení přes Docker či pečlivě projít, co se bude odinstalovávat.

```
$ sudo apt install aptitude
$ sudo aptitude install libpq-dev
```

Na první dotaz jsem dal nikoliv (`n`; starší balík by se nenainstaloval), na druhou nabídku ano (`y`). Zopakoval jsem instalaci v R a skončila úspěšně.

Zkusil jsem třířádkovou ukázku dle instalačního manuálu `coronavirus.john-coene.com`. (nutno zalistovat na webové stránce níž).

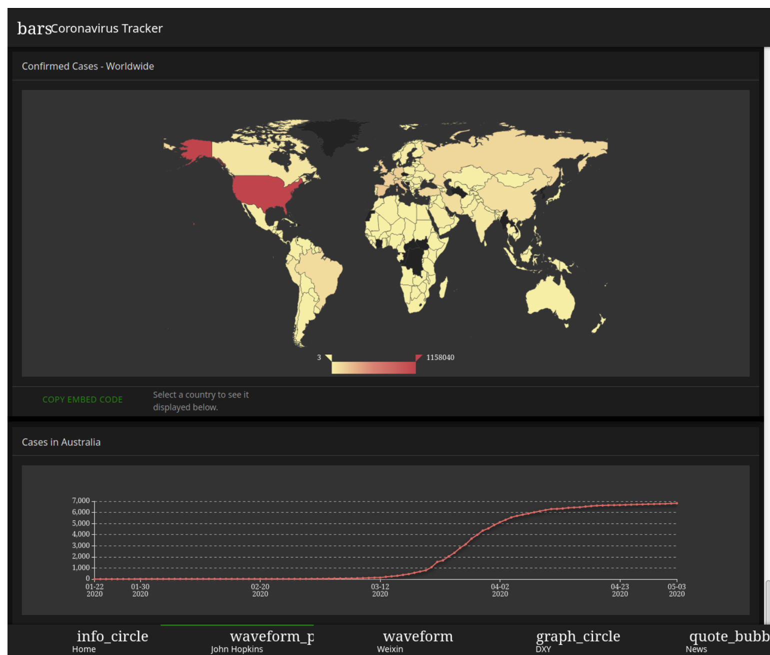
```
> library(coronavirus)
> virus<-crawl_coronavirus()
i Crawling data from John Hopkins
i Crawling data from Weixin
i Crawling data from DXY
> run_app(virus)
```

Pozn. Pokud bychom se dostali do konfliktu u příkazů, užijme:

```
> virus <- coronavirus::crawl_coronavirus()
> coronavirus::run_app(virus)
```

Ve webovém prohlížeči se mi otevřela vygenerovaná stránka, pokaždé na jiném portu. Radost byla veliká! Za pozornost stojí, že má být Johns Hopkins, to již někdo zahlásil autorovi k opravení.

Ve spodní části je pět záložek. Na druhé (waveform_path) v bloku *China* a *World* a čtvrté (graph_circle) v bloku *Cities* jako kdyby něco chybělo. Po nakliknutí do bloku se otevře detailní výpis. Vrátit zpět se dá přes xmark_circle_fill značku v pravém horním rohu. Design je trochu nezvyklý, ale musíme mít na paměti, že je to zaměřené na mobilní telefony a já to zkoušel na notebooku.



Detaily kolem dat je možné nalézt v levém horním rohu pod bars nebo menu. Z R lze server zavřít přes klávesovou kombinaci Ctrl+C.

Nyní dokumentace radí si nastavit `crontab` atd. Co mne zaujalo u stažení dat z DXY je, že se občas nezadařilo připojit. V rychlosti jsem nahlédl na server `https://education.rstudio.com/`, konkrétně na `dataio`.

Jakmile se podařilo na servery připojit, mohl jsem si proměnnou `virus` uložit a opětovně užívat. Rychlá pomůcka u experimentů bez nutnosti aktualizace dat.

```
> save(virus, file="virus.RData")
> load("virus.RData")
```

4. Hašovací klíč od newsapi.org v2

Mou pozornost zaujala poslední záložka se zprávou `No newsapi token.` To bych rád poléčil. Autor v dokumentaci radí:

```
> library(coronavirus)
> create_config()
```

V pozadí se ze šablony vytvoří soubor `_coronavirus.yml`, blok `database` je povinný, blok `newsapi` volitelný. To byl pro mne problém. Já jsem to chtěl obráceně. Nevadí.

Přes `https://newsapi.org/register` jsem se zaregistroval a získal hašovací klíč. Zahlédl jsem jejich novou knihovnu pro R `newsanchor`, my zůstaneme u autorem užité knihovny `newsapi`.

Zkusil jsem R podsunout hašovací klíč:

```
> library(newsapi)
> newsapi::newsapi_key("41e22e9efcf64b2a9354a796b99c43b8")
```

Ale ani touto cestou ani jinou přes editaci souboru `_coronavirus.yml` se mi to nepodařilo.

Prvně jsem nahlédl na zdrojové kódy v:

```
$ cd ~/R/x86_64-pc-linux-gnu-library/4.0/coronavirus
```

Narazil jsem hlavně na binární soubory rds, rdx a rdb. Nejsem expert, abych dokázal odpovědět, jestli by se soubory daly rozluštit a editovat.

Prozkoumal jsem zdrojové kódy přímo od autora:

```
$ git clone https://github.com/JohnCoene/coronavirus
```

Došel jsem k závěru, že bych musel zdrojové kódy upravit, zkompilovat atd. To je nad rámec této sváteční zprávy.

V souboru `coronavirus/inst/app/Dockerfile` jsem si ověřil, že skutečně knihovnu `newsapi` přebírá z GitHubu od uživatele `news-r`.

5. PostgreSQL v10+190

Říkal jsem si, když už se mi podařilo nainstalovat `libpq-dev`, dokáži i zbytek. Otevřel jsem komunitní tutoriál. Nainstaloval jsem PostgreSQL:

```
$ sudo apt install postgresql postgresql-contrib
```

A nejkratší možnou cestou jsem se pustil do dalších kroků. Vytvořil jsem v databázovém systému nového uživatele `testing` a novou databázi `testing`. Vynechávám krok vytvoření uživatele pod operačním systémem.

```
$ sudo -i -u postgres createuser --interactive
Enter name of role to add: testing
Shall the new role be a superuser? (y/n) y
$ sudo -u postgres createdb testing
```

Uživatel je bez hesla, to webové rozhraní nepřijme. Nastavil jsem nové heslo přes:

```
$ sudo -i -u postgres
$ psql
postgres=# ALTER USER testing WITH PASSWORD 'testing';
postgres=# \q
$ exit
```

Rychlokurz `psql`: `help` je základní nápověda, `\l` je výpis databází, `\c testing` je připojení k naší databázi, `\dt` je výpis tabulek, `\h` je seznam SQL příkazů, `\?` je seznam příkazů `psql` a `\q` ukončí běh programu. Verzálky u příkazů netřeba psát.

Vše zrealizované jsem zaznačil v `_coronavirus.yml`:

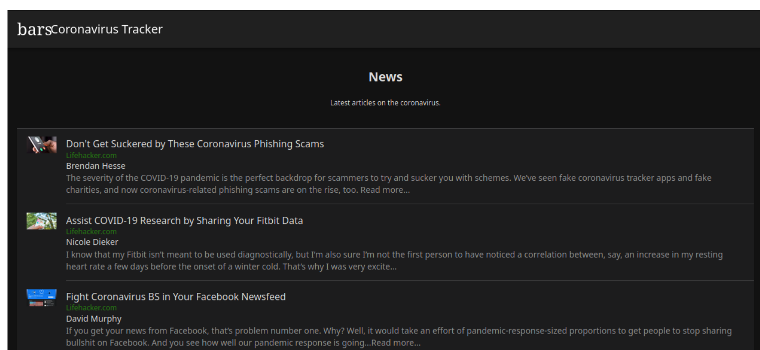
```
database:
  name: testing
  host: 127.0.0.1
  user: testing
  password: testing
newsapi:
  key: 41e22e9efcf64b2a9354a796b99c43b8
```

Když jsem opakoval tři řádky ukázkového spuštění v R, výpis se mi rozšířil o tyto dva řádky:

```
i Crawling news from newsapi.org
✔ Writing to database
```

V páté záložce mi vyběhly novinky, aktivovaný dotaz lze nalézt u autora v souboru `coronavirus/R/crawl.R`:

```
news <- newsapi::every_news("coronavirus OR covid", results = 100, language =
    "en", sort = "popularity")
```



Ověření funkčnosti můžeme zjistit i z tabulky `log`:

```
$ psql -h localhost -d testing -U testing
Password for user testing: testing
testing=# SELECT * FROM log;
```

Dostáváme přibližně takový výsledek:

```
         last_updated
-------------------------------
 2020-05-01 18:19:24.547171+02
(1 row)
```

Dle chuti lze dál bádat u surových dat, např.:

```
testing=# SELECT * FROM jhu WHERE country='Czechia';
testing=# SELECT * FROM jhu WHERE country='Slovakia';
testing=# \q
```

6. Bioconductor v3.11

Před dalším krokem si obvykle nastavuji plná práva u těchto adresářů:

```
$ cd /usr/lib/R
$ sudo chmod -R 777 site-library/
$ sudo chmod -R 777 library/
$ cd /usr/share
$ sudo chmod -R 777 R/
```

Za zmínku stojí, že manažer knihoven `biocLite` ustupuje a roli nahrazuje `BiocManager`. V R lze otestovat:

```
> install.packages("BiocManager")
> library(BiocManager)
> BiocManager::install()
> BiocManager::available()
```

Můžeme ověřit instalaci knihoven:

```
> BiocManager::valid()
   [...] "coronavirus", "echarts4r", "shinyMobile" [...]
Warning message:
0 packages out-of-date; 3 packages too new
```

To souhlasí, neb `coronavirus` byl instalován z GitHubu, nikoliv z CRANu.

Pro badatele stojí za pozornost obrazy pro Docker a Amazon (Amazon Machine Image, AMI). Je zde možnost instalovat vývojářské knihovny:

```
> BiocManager::install(version="devel")
```

7. Řešení konfliktu názvu knihovny v R

Na chvíli se ještě vraťme k řešení konfliktu stejného názvu knihoven. Na Stackoverflow zmiňují v principu dvě cesty.

Stáhnout si zdrojové soubory a nic neměnit:

```
$ cd /tmp
$ wget https://cran.r-project.org/src/contrib/coronavirus_0.1.0.tar.gz
$ R CMD INSTALL -l /tmp coronavirus_0.1.0.tar.gz
```

V R si pak volat jeden z příkazů a vybrat tak chtěnou knihovnu:

```
> # library(coronavirus)
> library(coronavirus, lib.loc="/tmp")
```

Když jsem zkoušel paralelně spustit i instalovanou knihovnu z GitHubu `coronavirus` odkomentováním prvního řádku, tak to neběželo. Tuším, že se jedná o bezpečnostní pojistku.

Druhá cesta je zasáhnout do souboru `DESCRIPTION`.

```
$ tar xvf coronavirus_0.1.0.tar.gz
$ mv coronavirus coronavirusRami
$ cd coronavirusRami
$ nano DESCRIPTION
```

První řádek upravit například na `Package: coronavirusRami`.

Volitelně upravíme i `MD5`, konkrétně první řádek za výpis:

```
$ md5sum -b DESCRIPTION
324f8275940bfa7fde376934c57a28ae *DESCRIPTION
```

Chtělo by to přejmenovat i další soubory na `coronavirusRami`, u této školní ukázky vynechávám. Zabalil jsem si zpět a už podsunul R:

```
$ cd ..
$ tar cvf coronavirusRami.tar.gz coronavirusRami/
$ R CMD INSTALL coronavirusRami.tar.gz
```

Ověřit funkčnost můžeme už přímo v R a lze si spustit oba konfliktní balíčky paralelně:

```
> library(coronavirus)
> ?coronavirus::crawl_coronavirus
> library(coronavirusRami)
> ?coronavirusRami::coronavirus
```

8. Pár tipů místo Závěru

Podobně jako jsou v R knihovny setříděné podle kategorií, viz Zobrazení úloh (CRAN Task Views), lze nahlédnout u RStudia na R Views s klíčovým slovem `covid-19`. K dnešnímu dni tam jsou tři záznamy: Some Select COVID-19 Modeling Resources, Simulating COVID-19 interventions with R a COVID-19 epidemiology with R.

Kdo by dal přednost odpočinku od R a PostgreSQL, nechť nahlédne na aktuální stavy kolem koronaviru na `https://www.twitch.tv/killars`.

Kontaktní adresa

Ing. Pavel Stříž, Ph.D., U Škol 940, Bučovice, okres Vyškov, 685 01, Česká republika,
E-mailová adresa: pavel@striz.cz

12. ročník
Žilina 2. – 4. 7.
2024
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSS Conf

OSSConf 2024: 117–124

POSTŘEHY NEJEN K SAZBĚ MATEMATIKY

PAVEL STŘÍŽ (CZ)

Abstrakt. Článek shrnuje novinky v TEXovém světě za poslední čas.

Klíčová slova. TEX, matematika.

NOTES NOT ONLY ON TYPESETTING MATH

Abstract. The article briefly introduces new and recently updated packages in the TEX world.

Keywords. TEX, mathematics.

1. Znaky a písma

Pravděpodobně nejrychlejší cesta v TEXLive (použitá verze 2020), jak získat rychlý přehled o dostupných znacích, symbolech a balíčcích, je otevřít si dokument symbols. Užíváme k tomu program `texdoc`. Případně užijte `ctan.org`.

```
$ texdoc symbols
```

Naopak přehled symbolů v Unicode lze vyčíst z balíčku xecjk.

```
$ texdoc xunicode-symbols
```

Takto by vypadala ukázka vysázení přes balíček halloweenmath.

```
$ texdoc halloweenmath
```

$$\bigskull_{i=1}^{n} H_i = H_1 \skull \cdots \skull H_n$$
$$\bigoplus_{i=1}^{n} H_i = H_1 \oplus \cdots \oplus H_n$$
$$\bigpumpkin_{i=1}^{n} H_i = H_1 \pumpkin \cdots \pumpkin H_n$$

$$\overbat{y} + \underbat{x} + z = 0$$

Nyní již vážněji. Za pozornost kolem písem s matematickými symboly stojí tyto odkazy. Z roku 2006 `http://mirrors.concertpass.com/tex-archive/info/Free_Math_Font_Survey/en/survey.pdf` a novější komunitní odpověď na serveru `https://tex.stackexchange.com/questions/425098`.

Řada TEXistů se již přímo či nepřímo setkala s balíčkem fontspec, za pozornost však stojí experimentální balíček unicode-math. Hlavní cíl je, aby se znaky daly zapisovat přímo, nikoliv přes TEXové příkazy, tedy místo `\alpha` zapisovat přímo α.

```
$ texdoc fontspec unicode-math
```

Zde je ukázka srovnání šesti písem: Latin Modern Math (L), XITS Math (X), STIX Math Two (S), TEX Gyre Pagella Math (P), DejaVu Math TEX Gyre (D) a Fira Math (F).

```
$ texdoc unimath-symbols
```

USV	L	X	S	P	D	F	Macro
U+003B1	α	α	α	α	α	α	\mupalpha
U+003B2	β	β	β	β	β	β	\mupbeta
U+003B3	γ	γ	γ	γ	γ	γ	\mupgamma
U+003B4	δ	δ	δ	δ	δ	δ	\mupdelta
U+003B5	ε	ε	ε	ε	ε	ε	\mupvarepsilon
U+003B6	ζ	ζ	ζ	ζ	ζ	ζ	\mupzeta
U+003B7	η	η	η	η	η	η	\mupeta
U+003B8	θ	θ	θ	θ	θ	θ	\muptheta

1.1. Nové přírůstky: úplná sada

Rodiny písem Latin Modern a TEX Gyre se stávají standardy. V linuxovém světě mají svou oblibu projekty DejaVu a Libertinus (Libertine+Biolinum), které mají svá matematická písma. Vedle již běžných emoji se objevují ligatury pro programátory, viz FiraCode.

Mezi nové přírůstky na `ctan.org` u matematických písem počítáme od roku 2018 STIX Two (nástupce písma STIX a pokračování matematického písma XITS), GFS Neohellenic (založeno na písmu New Hellenic) a Fira Math (založeno na písmu Fira Go) a od roku 2019 písma Garamond Math (založeno na písmu EB Garamond) a Erewhon Math (založeno na rodinách písem Utopia, Heuristica a Erewhon).

1.2. Nové přírůstky: neúplná sada

To ale není vše, co balíček unicode-math umí. Některá písma, např. Berenis ADF Pro či Neo Euler, nejsou dokončená, nemají celé řezy či mají závady. Můžeme užít `range` a vybrat si jen bloky z písma.

Zde je ořezaná ukázka preambule dokumentu, jak by se to dalo vyřešit.

```
\unimathsetup{math-style=upright}
\setmainfont{CMU Concrete}
\defaultfontfeatures{Scale=MatchLowercase}
\setmathfont{Libertinus Math}
\setmathfont[range={"0000-"0001,"0020-"007E,
```

```
  "00A0,"00A7-"00A8,"00AC,"00AF,"00B1,"00B4-"00B5,"00B7,
  % řada dalších
  "1D6DF,"1D6E1,"1D7CE-"1D7D7
  }]{Neo Euler}
\setmathfont[range=up/{greek,Greek}, script-features={},
    sscript-features={}]{Neo Euler}
\setmathfont[range=up/{latin,Latin,num}, script-features={},
    sscript-features={}]{Neo Euler}
```

Nabízí se ještě jedno užití, a to vybrat blok s okrasnými či atypickými znaky. Zde je ukázka u vzpřímeného znaku integrálu.

```
\setmainfont{XITS}
\setmathfont{XITS Math}
\setmathfont[range={"222B-"2233,"2A0B-"2A1C},StylisticSet=8]{XITS Math}
```

Theorem 1 (Residue theorem). *Let f be analytic in the region G except for the isolated singularities $a_1, a_2, \ldots, a_m$. If γ is a closed rectifiable curve in G which does not pass through any of the points a_k and if $\gamma \approx 0$ in G, then*

$$\frac{1}{2\pi i} \int_\gamma f\left(x^{\mathrm{N}\in\mathbb{C}^{N\times 10}}\right) = \sum_{k=1}^{m} n(\gamma; a_k) \operatorname{Res}(f; a_k).$$

Theorem 2 (Maximum modulus). *Let G be a bounded open set in $\mathbb{C}$ and suppose that f is a continuous function on G^- which is analytic in G. Then*

$$\max\{\, |f(z)| : z \in G^- \,\} = \max\{\, |f(z)| : z \in \partial G \,\}.$$

First some large operators both in text: $\iiint\limits_{Q} f(x, y, z)\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z$ and $\prod_{\gamma\in\Gamma_{\tilde{C}}} \partial(\tilde{X}_\gamma)$; and also on display

$$\iiiint\limits_{Q} f(w, x, y, z)\,\mathrm{d}w\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z \leq \oint_{\partial Q} f'\left(\max\left\{ \frac{\|w\|}{|w^2 + x^2|}; \frac{\|z\|}{|y^2 + z^2|}; \frac{\|w \oplus z\|}{|x \oplus y|} \right\}\right).$$

1.3. Rozšíření IBM Plex v roce 2021

Písmo, od počátku počítačů, se stalo nástrojem bojů velkých IT firem. IBM představilo rodinu písem Plex a 21. 4. 2020 oznámil Mike Abbink rozšíření matematiky někdy v roce 2021. Již nyní se však dá otestovat vzorky písem uvnitř matematiky za pomocí balíčků plex a mathastext. Druhý balíček umožňuje v matematickém režimu přebrat znaky z běžného písma.

Upravená preambule by vypadala takto:

```
\usepackage{unicode-math}
\usepackage{mathastext}
\usepackage{plex-serif}
```

Zaujal mě balíček mathastext samotný. Zde je, v dnešní době, netradiční ukázka, kdy v matematickém režimu užijeme proporcionální písmo rodiny Latin Modern.

Let (X, Y) be two functions of a variable a. If they obey the differential system $(VI_{\nu,n})$:

$$a\frac{d}{da}X = \nu X - (1 - X^2)\frac{2na}{1-a^2}\frac{aX+Y}{1+aXY}$$
$$a\frac{d}{da}Y = -(\nu+1)Y + (1 - Y^2)\frac{2na}{1-a^2}\frac{X+aY}{1+aXY}$$

then the quantity $q = a\frac{aX+Y}{X+aY}$ satisfies as function of $b = a^2$ the

Pro badatele doporučuji nahlédnout na ukázky na webové stránce `http://jf.burnol.free.fr/showcase.html`. Balíček nám umožňuje užít prakticky libovolné písmo. Zde je jedna vizuální ukázka s ručně psaným písmem Chalkduster pomocí XƎLaTeXu.

```
\usepackage[no-math]{fontspec}
\setmainfont[Mapping=tex-text]{Chalkduster}
\usepackage[defaultmathsizes]{mathastext}
```

The special case $A = C$, $B = D$, gives

$$\begin{vmatrix} Ad(u) & Bd(x) \\ Ad(v) & Bd(y) \end{vmatrix}_{2n\times 2n} = \det(A)^2 \det_{1\leqslant i,j\leqslant n}((u_i y_j - v_i x_j)(A^{-1}B)_{ij}) \tag{6}$$

Let $W(k)$ be the Vandermonde matrix with rows $(1\dots 1)$, $(k_1 \dots k_n)$, $(k_1^2 \dots k_n^2)$, ..., and $\Delta(k) = \det W(k)$ its determinant. Let

$$K(t) = \prod_{1\leqslant m\leqslant n}(t - k_m) \tag{7}$$

1.4. Ze světa pravolevé sazby matematiky

Hans Hagen zmiňuje tyto, pro nás Evropany extrémní, ukázky sazby matematiky v dokumentu *Fonts out of ConTEXt*.

```
$ texdoc fonts-mkiv
```

$$\sqrt{\frac{\text{ف}^2}{\text{ب}4}} \qquad \left(\sqrt[2]{155}\right)$$

$$\left[{}^3666 \int_{123}^{55}\right] \qquad \left[{}^3666 \int_{123}^{55}\right]$$

$$4 < 6 > 5 \qquad \left\{{}^3666 \sum_{123}^{55}\right\}$$

$$7 \geq 6 \leq 5$$

2. Proměnlivá velikost

Proměnlivá velikost znaků v matematice není žádná novinka, ať už se podíváme na AMSTEX nebo rozšíření mathtools.

```
$ texdoc amslatex mathtools
```

Mou pozornost zaujal měněný úhel sklonu u odmocniny z přednášky Hanse Hagena z roku 2018.

```
$ texdoc bachotex-2018-fonteffects
```

$$2 \times \sqrt{\frac{\sqrt{\frac{\sqrt{2}}{\sqrt{2}}}}{\sqrt{\frac{\sqrt{2}}{\sqrt{2}}}}}$$

Do popředí se dostává podpora síly linky, byť to vypadá, že typografická revoluce přes formát cff firmy Adobe nenastane.

```
$ texdoc bachotex-2017-variablefonts
```

Adobe Variable Font Prototype (cff)

extralight 0/0	It looks like this!
light 150/0	It looks like this!
regular 394/0	It looks like this!
semibold 600/0	**It looks like this!**
bold 824/0	**It looks like this!**
black high contrast 1000/100	**It looks like this!**
black medium contrast 1000/50	**It looks like this!**
black 1000/0	**It looks like this!**

3. Užití barvy

Za zajímavost uvádím, že první barevná kniha je datována do roku 1633, autor je Hu Zhengyan (胡正言, 1584–1674). Jestli proběhne revoluce v písmech z ttf/otf na barevná, viz www.colorfonts.wtf, to se ještě uvidí.

Obarvit si proměnné můžeme poloručně se značkami, $x^2 + y^2 = z^2$, takto:

```
\def\barvaR{\color{red}}
\def\barvaG{\color{green}}
\def\barvaB{\color{blue}}
\def\barva{\color{black}}
$\barvaR x^2\barva+\barvaG y^2\barva=\barvaB z^2$
```

Zajímavý je nápad nezařazovat do textů vlnku, ale nedělitelné jednoznakové předložky a spojky mít v textu bez ní, viz balíčky encxvlna od Petra Olšáka a Zdeňka Wagnera, xevlna od Zdeňka Wagnera a luavlna od Michala Hofticha a Mira Hrončoka.

```
$ texdoc encxvlna xevlna luavlna
```

Zkusíme si tuto úvahu aplikovat u obarvení proměnných.

```
$ texdoc luatex about
```

Zde je jedna víceméně nepraktická ukázka představující možnosti. Spouštíme `lualatex obarveni.tex`, proměnné nemají značky, přidají se za běhu.

```
\documentclass[a4paper]{article}
\usepackage{luacode}
\usepackage{xcolor}
\def\zpet{\color{black}}
\def\barvax#1{\color{red}#1\zpet}
\def\barvay#1{\color{green}#1\zpet}
\def\barvaz#1{\color{blue}#1\zpet}
\begin{luacode*}
function obarvi (incoming)
incoming=unicode.utf8.gsub(incoming, "%$?%$.-%$?%$", function(s)
  print("Našel jsem matematiku: "..s.."\n")
  s=unicode.utf8.gsub(s, "x^?2?", "\\barvax{%1}")
  s=unicode.utf8.gsub(s, "y^?2?", "\\barvay{%1}")
  s=unicode.utf8.gsub(s, "z^?2?", "\\barvaz{%1}")
  return s
end) -- úprava incoming
return incoming
end -- function obarvi
luatexbase.add_to_callback("process_input_buffer",obarvi,"obarvi")
\end{luacode*}
\begin{document}
Text před výrazem s $x$, $y$ a $z$.
$$x^2+y^2=z^2 \to x^2=z^2-y^2 \to x=\pm\sqrt{z^2-y^2}$$
Text za výrazem s $x$, $y$ a $z$.
\end{document}
```

Text před výrazem s x, y a z.

$$x^2+y^2=z^2 \to x^2=z^2-y^2 \to x=\pm\sqrt{z^2-y^2}$$

Text za výrazem s x, y a z.

Kdo by se rád podíval na jiné úpravy textů pomocí LuaTEXu, doporučuji jako startovní bod balíčky chickenize a typewriter.

```
$ texdoc chickenize typewriter
```

4. Kresba znaků

Mezi první pokusy s barevnými písmy v digitální době řadím práce tvůrce písem Manfreda Kleina a Edwarda R. Tufteho. Tufte v *Envisioning Information* (1990, str. 33) uvádí příklad u vizualizace trička: mít jen obrysy a měnit barvu výplně.



V TEXovém světě je jeden z nejvýraznějších nápadů v této oblasti v sazbě šachu od Ulrike Fischer. Zde je náhled z balíčků chessboard a xskak. Jedná se o přípravu znaků (šachových figurek a pozadí na šachovnici) vrstvením kreseb (obr. vlevo). Kresby jsou uloženy v písmu, jejich úprava je komplikovanější.

```
$ texdoc chessboard xskak
```

Příchod Ti*k*Zu technicky s editací pomohl. Lze nahlédnout na nové balíčky bclogo, tikzsymbols, tikzducks (princip znázorněn na obrázku vpravo), tikzlings (tučňák v závěru), tikzmarmots a tikzpeople.

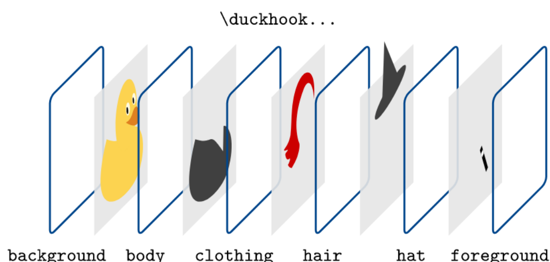
```
$ texdoc bclogo tikzsymbols tikzducks tikzlings tikzmarmots tikzpeople
```

Encoding LSBC3

Layer:	fieldmask	field	piecemask	piece	result
WhiteSquare					
BlackSquare					
WhiteOnWhite					
WhiteOnBlack					
BlackOnWhite					
BlackOnBlack					

At last there is the char for the king:

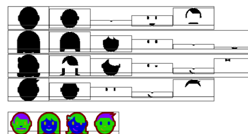
So



Hans Hagen poukazuje na sazbu emoji v přednášce z roku 2017.

```
$ texdoc bachotex-2017-emoji
```

seguiemj	default		
	overlay		
emojionecolor-svginot	default		
	svg		
emojionemozilla	default		
	overlay		
applecoloremoji	default		
	bitmap		



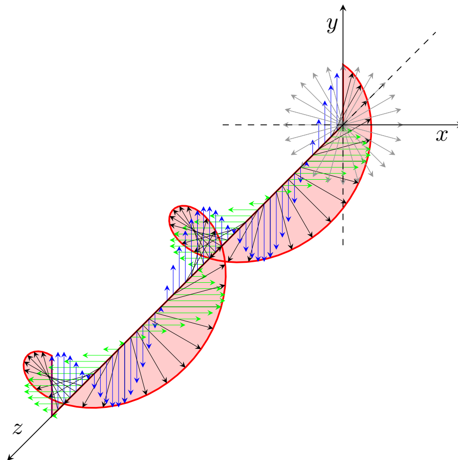
5. Popisky grafu

Na závěr si ukažme, jak lze snadno popsat a vykreslit 2D, ale i 3D graf, aniž bychom užili rozsáhlý balíček pgfplots. Zůstaneme jen u Ti*k*Zu.

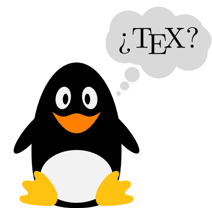
```
$ texdoc pgfplots tikz
```

U příkazu `\draw` místo `(x,y)` pracujeme s `(x,y,z)`, zde je ukázka z mé zahrádky jako odpověď na `tex.stackexchange.com/questions/167137`.

```
\documentclass[a4paper]{article}
\usepackage{tikz}
\usetikzlibrary{intersections}
\begin{document}
\tikzset{malstyle/.style={->, >=stealth, line width=0.2pt},
  malarrow/.style={->, >=stealth}}
\begin{tikzpicture}
\draw [name path=Ewave] [red, thick, ->, fill, fill opacity=0.2] (0,0,0) -- plot [domain=0:12.5664,
     samples=100] ({sin(\x r)}, {cos(\x r)}, \x) -- (0,0,12.5664) --cycle;
\foreach [ evaluate={\xpos=sin(\zpos*180/pi); \ypos=cos(\zpos*180/pi);} ]
   \zpos in {0, 0.2618, ..., 12.5664} {% Začátek \foreach...
  \draw[malstyle, black] (0,0,\zpos) -- (\xpos, \ypos, \zpos);
  \draw[malstyle, black!40] (0,0,0) -- (\xpos, \ypos, 0);
  \draw[malstyle, green] (0,0,\zpos) -- (\xpos, 0, \zpos);
  \draw[malstyle, blue] (0,0,\zpos) -- (0, \ypos, \zpos);
  }% Konec \foreach...
\draw [malarrow] (0,0,0) -- (0,0,14.5) node[xshift=5, yshift=15] {$z$};
\draw [malarrow] (0,0,0) -- (0,2,0) node[xshift=-5, yshift=-10] {$y$};
\draw [malarrow] (0,0,0) -- (2,0,0) node[xshift=-10, yshift=-5] {$x$};
\draw[dashed] (0,0,0)--(-2,0,0) (0,0,0)--(0,-2,0) (0,0,0)--(0,0,-4);
\end{tikzpicture}
\end{document}
```




```
\begin{tikzpicture}
\penguin[think={?`\TeX?},bubblecolour=gray!30!white]
\end{tikzpicture}
```



Kontaktní adresa

Ing. Pavel Stříž, Ph.D., U Škol 940, Bučovice, okres Vyškov, 685 01, Česká republika, *E-mailová adresa*: `pavel@striz.cz`

12. ročník Žilina 2. – 4. 7. 2024
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
osscconf.soit.sk
OSS Conf
OSSConf 2024: 125–132

RAYLIB + R = RAYLIB × R = RAYLIBR: AHOJ, SVĚTE!

PAVEL STŘÍŽ (CZ)

Abstrakt. Článek stručně odkazuje na instalaci Raylibu (prostředí pro 2D a 3D grafiku napsaný primárně v programovacím jazyce C), ale též na mnohem důležitější RaylibR. Jedná se o most mezi Raylibem a výpočetním prostředí R. Užívá k tomu eRkový balíček `Rcpp`. Byl to do určité míry boj, který však stál za to.

Klíčová slova. Raylib, R, RaylibR, Rcpp.

RAYLIB + R = RAYLIB × R = RAYLIBR: HELLO, WORLD!

Abstract. The article briefly refers to an installation of Raylib (an environment for 2D and 3D graphics written primarily in the C programming language), but most importantly to an installation of RaylibR. That's a wrapper of the Raylib environment for the R environment written with the help of the `Rcpp` package. It was a bit of struggle, but it paid its dividends in the end.

Keywords. Raylib, R, RaylibR, Rcpp.

1. Jak jsem do toho spadl

Musím se přiznat, že nejsem Céčkař (nedejbůh Cépépéčkař, zatím) ani eRkař, aspoň se tak necítím, učím se za pochodu. Během programování sudoku s překryvy (angl. multi-sudoku puzzle; multi-grid sudoku; overlapping sudoku) jsem měl nápad udělat si 2D a 3D vizualizaci ze zadání do řešení. Ve stylu arkádovek 80. let, či aspoň nějak podobně. TEX umí hodně věcí, ale tohle není jeho doména, resp. má doména v TEXu, tak jsem hledal alternativu.

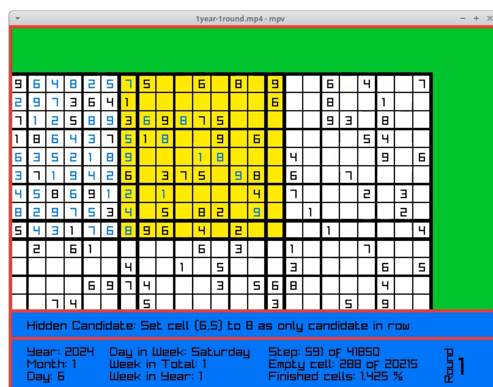
Rudolf Blaško má hezké výstupy přes Asymptote, ale na ten jsem rezignoval, potřeboval jsem ještě větší flexibilitu (myšleno programátorskou svobodu). Na vlastní prostředí jsem to také neviděl.

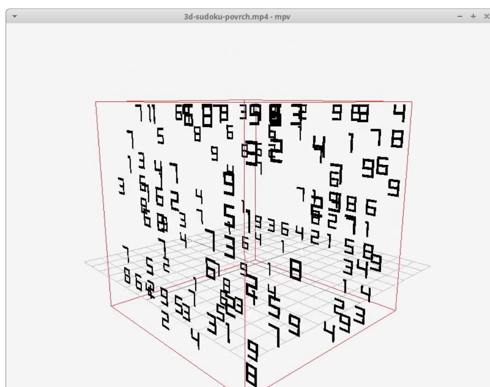
Zkusil jsem tedy řadu existujících grafických prostředí pro C, moc se mi líbí (od X11, FLTK, ImGui, GTK+ přes Qt, Cairo, SFML až po SDL2), objevil jsem i malé projekty typu Olive.c a v tom čase jsem narazil na Raylib od Ramona Santamarie. Užívá GLFW a OpenGL. Zrovna na YouTube slavil 10 let existence projektu. Objev byl učiněn hlavně díky tomuto videu `https://www.youtube.com/watch?v=0To1aYglVHE`, kde uživatel Tsoding Daily zkoumá uložení projektu do videa.

Prošel jsem webové stránky a zjistil, že by to mělo běžet na většině operačních systémů, platforem a ve všech známějších programovacích jazycích. Ani jsem

nepomyslel na Python, hned jsem skočil po nativní verzi v C. Za pomoci seznamu příkazů, `https://www.raylib.com/cheatsheet/cheatsheet.html`, a několika videí na YouTube, to začalo běžet.

Přikládám dvě ukázky. Není tam běhající či létající panáček či panenka, bylo potřeba se držet svých mantinelů. Sudoku generuje Lua, to jsou mé experimenty do Vánoce 2023, a Python3, to jsou nové pokusy od povánočních svátků dál. 2D rozkres krychle zadání a řešení níž.

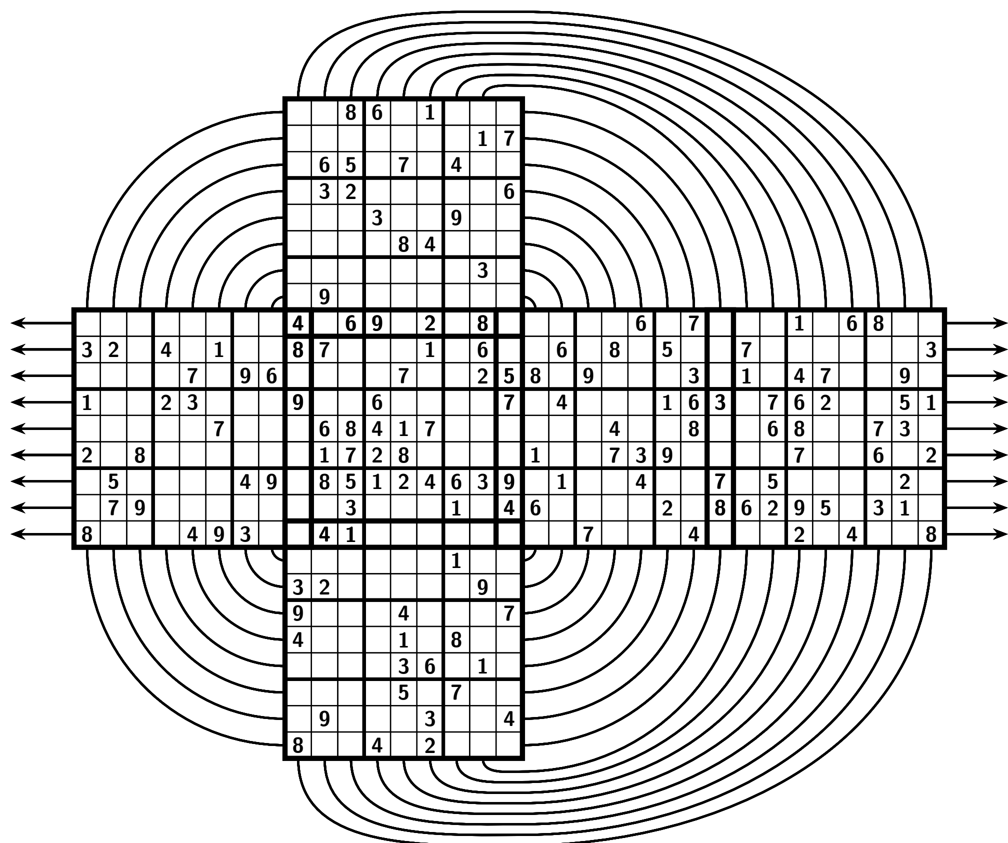




								7	4	8	6	9	1	3	5	2																
								9	2	3	8	4	5	6	1	7																
								1	6	5	2	7	3	4	9	8																
								5	3	2	7	1	9	8	4	6																
								8	1	4	3	2	6	9	7	5																
								6	7	9	5	8	4	1	2	3																
								2	8	1	4	6	7	5	3	9																
								3	9	7	1	5	8	2	6	4																
7	9	1	5	8	6	2	3	4	5	6	9	3	2	7	8	1	4	9	3	5	6	8	7	2	5	3	1	9	6	8	4	7
3	2	6	4	9	1	5	7	8	7	2	5	4	1	9	6	3	7	6	4	8	2	5	1	9	7	4	5	8	2	1	6	3
5	8	4	3	7	2	9	6	1	3	9	8	7	6	4	2	5	8	2	9	1	7	4	3	6	1	8	4	7	3	2	9	5
1	6	7	2	3	5	8	4	9	2	4	6	5	3	8	1	7	5	4	2	9	8	1	6	3	8	7	6	2	9	4	5	1
9	4	5	8	1	7	6	2	3	6	8	4	1	7	5	9	2	9	3	6	4	1	7	8	5	2	6	8	4	1	7	3	9
2	3	8	9	6	4	7	1	5	1	7	2	8	9	3	4	6	1	8	5	7	3	9	2	4	9	1	7	3	5	6	8	2
6	5	3	1	2	8	4	9	7	8	5	1	2	4	6	3	9	2	1	8	6	4	3	5	7	4	5	3	1	8	9	2	6
4	7	9	6	5	3	1	8	2	9	3	7	6	8	1	5	4	6	7	1	3	5	2	9	8	6	2	9	5	7	3	1	4
8	1	2	7	4	9	3	5	6	4	1	3	9	5	2	7	8	3	5	7	2	9	6	4	1	3	9	2	6	4	5	7	8
								5	8	9	6	2	7	1	4	3																
								3	2	7	1	8	4	6	9	5																
								9	1	2	5	4	8	3	6	7																
								4	6	3	7	1	9	8	5	2																
								7	5	8	2	3	6	4	1	9																
								2	3	4	9	5	1	7	8	6																
								1	9	6	8	7	3	5	2	4																
								8	7	5	4	6	2	9	3	1																

2. Vzorek z vlastní zahrádky

Na této straně je řešení, na další straně zadání 3D sudoku (povrch krychle, tedy 6 sudoku), rozkresleného na ploše se vztahy. Vztahy jsou komplikovanější (hrany sdílí dvě sudoku, rohy dokonce tři), ale princip vzniku je podobný.



3. Instalace knihovny Raylib

Co byl pro mne problém byla instalace, jak pracuji záměrně na starém notebooku, protože co mi běží na něm, poběží i na rychlejších strojích, toť základní idea. Hlavní zádrhel byl, že nemám OpenGL verzi 3.3, ale nižší (v2.1), tedy se to při instalaci knihovny a ukázek musí nastavovat. Svou verzi zjistíte na Linuxu přes knihovnu `mesa-utils`:

```
$ glxinfo | grep "OpenGL version"
```

V principu jsem následoval návod, přes `sudo make install` mám knihovny pro Desktop verzi, lokálně pak pro Web verzi (tu používám minimálně). Tím nenápadně naznačuji, že Raylib umí generovat své výstupy pro webové prohlížeče – za pomocí WebAssembly.

Když mé náhledy videoukázek viděl Aleš Kozubík vznesl dotaz, jestli by nebylo možné Raylib aktivovat v R, že by se jim tam něco takového hodilo. Na první dvě kola jsem R v seznamu jazyků na Raylibu na mobilu přehlédl, užil jsem RaylibRS, jak je v logu písmeno R, to bylo však pro Rust. Pak jsem si očistil brýle a na normálním monitoru to našel! Ve vyhledávači jsem se přímo zeptal na RaylibR.

4. Instalace RaylibR

Instalovat Raylib sólově netřeba, jen si stačí pobrat závislé balíčky.

```
sudo apt install build-essential git cmake libasound2-dev
  libx11-dev libxrandr-dev libxi-dev libgl1-mesa-dev
  libglu1-mesa-dev libxcursor-dev libxinerama-dev
```

Zkusil jsem v R:

```
> install.packages("Rcpp")
> install.packages("remotes")
> remotes::install_github("jeroenjanssens/raylibr")
```

Ale instalace mi zkolabovala, neb nemůže najít `R_ext/Error.h`. Začal mi trochu boj. Podezříval jsem `Rcpp`, ale nebylo to tím. Na GitHubu v *Open Issues* je tento problém jako jediný otevřený (k datu 17. 2. 2024), ale tipy na řešení mi nezabraly, tak jsem zkoumal. Našel jsem dvě cesty.

První, byť druhá v pořadí vzniku, je tato. Soubor mi v počítači existuje, jen v trochu jiné složce. Jestli je to dané tím, že jsem R instaloval z linuxového repozitáře versus přímo verzi ze CRANu, nevím. V ~`/.bashrc` jsem si přidal:

```
export C_INCLUDE_PATH=$C_INCLUDE_PATH:/usr/share/R/include
```

Je to vlastně snadné, když člověk ví, co hledat. Má první cesta byla o něco bojovnější. Stáhl jsem si z GitHubu RaylibR. Rozbalil jsem `raylibr-main.zip`. Nic s termínem `Error.h` jsem nenašel. Po hlubším zkoumání jsem zjistil, že Raylib je přítomen ve složce `inst/` jako `raylib-4.0.0-modified. tar.gz`, rozbalil jsem jej, a pak už to bylo „snadné“. Ve složce `src/` je `Makefile`, do kterého jsem zasáhl. V mém případě na řádku 207 odkomentováním:

```
GRAPHICS = GRAPHICS_API_OPENGL_21
```

Ale co je důležitější, přidal jsem na řádku 379 na konec parametr na mou cestu s žádaným souborem, do tvaru:

```
INCLUDE_PATHS = -I. -Iexternal/glfw/include -Iexternal/glfw/deps/mingw
  -I$(R_HOME)/include -I/usr/share/R/include
```

Zabalil jsem složku `raylib-4.0.0-modified` do `tar.gz`. A chybí už jen jeden krok, jít o dvě úrovně výš a zabalit složku `raylibr-main` do souboru `raylibr-main.tar.gz`. Pozor, R (zatím) neumí pracovat se `zip` soubory.

V R už to pak bylo přímočaré:

```
> install.packages("raylibr-main.tar.gz",repo=NULL)
```

Radost to byla veliká, byť takto psané mi to přijde vše jasné, stručné a logické. Své dva objevy jsem připsal autorovi na GitHub, je velká šance, že i tato chyba bude uzavřena.

5. Několik ukázek

Seznam ukázek získáme v R přes:

```
> library(raylibr)
> demo(package="raylibr")
```

Zkusit si jednu z nich, *Hello, World!*, můžeme takto:

```
> demo("helloworld",package="raylibr") # nebo
> demo(raylibr::helloworld) # či dokonce jen demo(helloworld)
```

Alternativa zobrazení dostupných ukázek v R je:

```
> help.start()
  Vybrat: Packages --> raylibr --> Code demos.
```

Ukončení R je příkazem `q()` nebo `quit()`. Volíme ano či ne (y/n), chceme-li si uložit pracovní prostředí, nechceme-li práci zatím přerušit volíme pokračovat (c).

V krátké přednášce na YouTube Jeroen Janssens, autor RaylibR, zmínil své dva vzorky – `raylibr::stroop` v úvodu povídání a `raylibr::beatbox` v jejím závěru.

A mj. ty ukázky jsou interaktivní, takže toho hada si můžete zahrát, v tom bludišti skutečně můžete chodit ap. Více ukázek (přes sto) hledejte přímo na stránkách Raylibu, v RaylibR jich je „jen“ jedenáct. Po instalaci jsem `*.R` našel v `~/R/x86_64-pc-linux-gnu-library/4.3/raylibr/demo/`.

Pro úplnost článku uvádím i zdrojový kód pro R.

```
library(raylibr)
init_window(600,400,"R & Raylib: Hello, World!")
while (!window_should_close()) {
  alpha<-abs(sin(get_time()))
  begin_drawing()
  clear_background("black")
  draw_circle(300,200,seq(150,10,by=-10),c("red","white"))
  draw_text(c("Hello,","World!"),225,c(120,220),64,fade("black",alpha))
  draw_fps(10,10)
  end_drawing()
  }
```

```
close_window()
```

Soubor se jmenuje `helloworld.R` a spustíme si jej z příkazového řádku přes:

```
$ Rscript helloworld.R
```

Okno s Raylibem se zavře přes klávesu Escape.

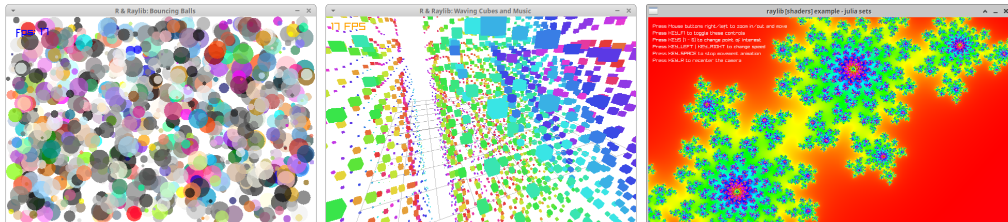
6. Závěrem

Asi by byla pro mne zajímavá výzva zkusit své „arkádové“ pokusy (Raylib+C) překlopit do Raylib+R a podat o zkušenostech s konverzí zprávu, ale nejdu do toho. Jsou další úkoly. Možná by šel udělat automat na konverzi, nechávám jako otevřený problém.

Musím se přiznat, že to byl pro mne boj, i Raylib i poté RaylibR, ale zvítězil jsem. Těším se, až nahodím silnější stroj s OpenGL v3.3 a zkusím si instalace ještě jednou. O tom možná pár slov na konferenci OSSConf, do termínu uzávěrky sborníku testy skoro určitě nestihnu. Hlavně se těším na ukázky v Raylibu (speciálně složka `shaders/`), ne všechny mi plně jely. Možná to bude tím, že pro v2.0 a v2.1 není podpora ve složce `examples/shaders/ resources/shaders/`, ale jen pro starší OpenGL v1.0, v1.2, a pak až pro novější v3.3…

Držím s instalacemi palce, nenechte se odradit, pokud vám něco na prvních deset pokusů nejede!

Zde jsou mé oblíbené ukázky: `raylibr::balls` za 2D a `raylibr::cubes` za 3D grafiku, už běžící přes RaylibR.



7. Post Scriptum

Mohu shrnout zkušenosti z novějšího notebooku s Xubuntu a OpenGL v4.3. Instalace Raylibu byla vzorná dle návodu, předvolená je verze 3.2 OpenGL, nebylo tedy potřeba zasahovat. Po přidání zmíněného řádku do `~/.bashrc` mi šla i instalace RaylibR přímo v R. Poznámka: v pozadí RaylibR bere Raylib verze 4.0, je však již verze 5.0. Je stále co zlepšovat!

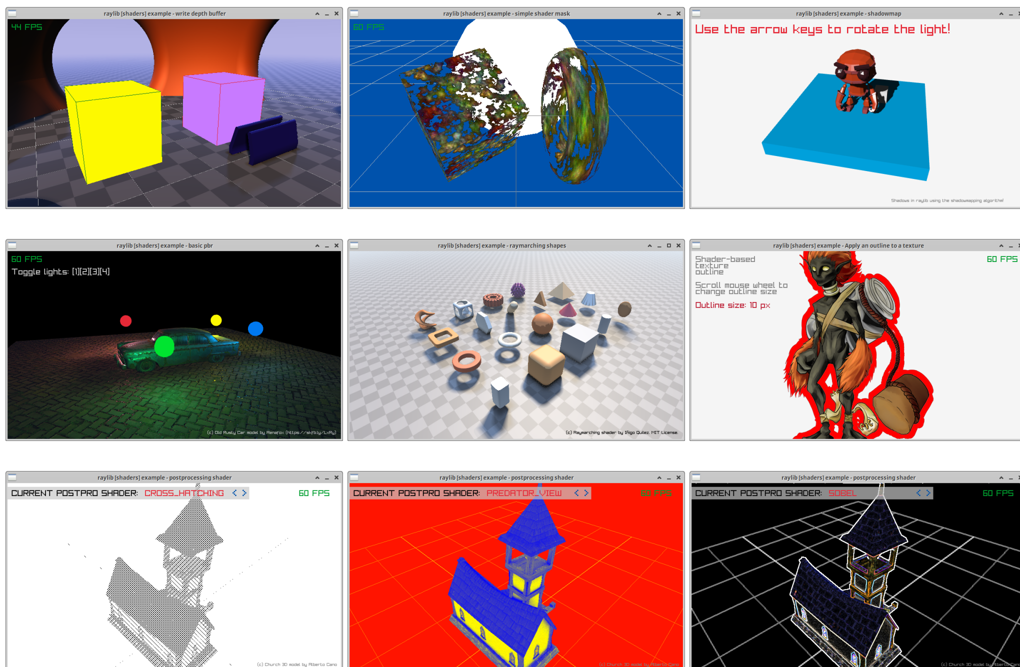
Zde je drobný dávkový soubor, který vám spustí všechny zkompilované ukázky, jednu za druhou. Napočítal jsem jich 146+1 šablona. Soubor mám uložený ve složce `raylib-master/`.

```
# Prázdný řetězec pro všechny, či vybranou složku z:
# audio core models others shaders shapes text textures
```

```
malkde=""
cd examples/$malkde
for soubor in `find . -type f -executable | sort`; do
echo "Spouštím $soubor..."
./$soubor # tichý režim: ./$soubor 1>/dev/null 2>/dev/null
done
```

Ukázky jsou výtečné, minimálně animovaná `raylibr::julia` v R či ukázka `shaders_julia_set` v Raylibu je neokoukaná (obrázky výš, vpravo).

Pro potěšení oka, z těch dalších, které mi v Raylib v5.0 na staříčkovi nejely, zmíním ze složky `examples/shaders/`, prefix `shaders_`, tyto ukázky: `write_depth`, `simple_mask`, `shadowmap` na prvním řádku, `raymarching`, `basic_pbr`, `texture-_outline` na 2. řádku a na 3. řádku z `postprocessing`, speciálně cross_hatching, predator_view a sobel.



8. Symbolická poznámka

Název článku má být „pomsta“ za ten boj s instalacemi. Při $x + y = x \cdot y$, tedy $y = x/(x-1)$ či $x = y/(y-1)$, ani jeden z programů nemůže být sám o sobě jedničkou. Raylib je skvělý na hry, ale o světě R neví nic. Naopak R má na vizualizaci tohoto typu ještě dost velké rezervy. Ale oba programy mohou být zároveň nula, to byl ten případ, kdy se nedařilo nainstalovat Raylib ani RaylibR, ze začátku to nešlo a nešlo…

Držím palce, ať je vždy $x + y > 1$! Třeba dvě dvojky, nemusí to být hned jedničky s hvězdičkou.

Kontaktní adresa

Ing. Pavel Stříž, Ph.D., U Škol 940, Bučovice, okres Vyškov, 685 01, Česká republika,
E-mailová adresa: `pavel@striz.cz`

AKO PROGRAMOVAŤ S AI

BIŇAS Miroslav (SK)

S príchodom četovacích botov umelej inteligencie, ktorých najznámejším predstaviteľom je ChatGPT, sa stretávame s množstvom otázok a pochybností, čo všetko tieto technológie dokážu priniesť a ovplyvniť. Rovnako je tak tomu aj v prípade vzdelávania. Na jednej strane pochybnosti o tom, ako sa zmení rola učiteľa a na druhej vidina hrozby v možnostiach podvádzania študentmi. Ako sa teda zmení výučba informatiky a programovania s príchodom četovacích botov? A ako ich môžu pre svoju prácu využívať aj samotní učitelia?

V tomto príspevku sa pozrieme na možnosti využitia četovacích botov práve učiteľmi. Ukážeme si, aké úskalia so sebou obnášajú zadania v podobe genericky známych problémov, ako je možné pomocou promptov postupne dôjsť k riešeniu zadanej úlohy, ale aj to, ako kreatívny a inšpiratívny môže byť četbot pri vymýšľaní opisov a sprievodných príkladov vysvetľujúcich preberanú problematiku.

CHYTRÁ KATEDRA

BIŇAS Miroslav (SK)

S rastom popularity problematiky Internetu vecí (IoT) sa IoT pomaly dostáva aj do osnov na stredných a vysokých školách. Tu však nastáva problém, ktorý súvisí s výberom tém pre takéto kurzy a otázka, ako vlastne problematiku IoT učiť. Stále totiž prevláda prístup, podľa ktorého robiť IoT znamená programovať mikrokontroléry a pripájať k nim elektronické senzory a akčné členy.

IoT však zahŕňa výrazne viac tém, ako len programovanie samotných mikrokontrolérov. Problematika IoT je totiž aj o dátach, ktoré si zariadenia vymieňajú, o ich štruktúre, ukladaní, o ich analyzovaní a vizualizovaní. Rovnako tak je aj o bezpečnosti v každej jednej časti oblasti, o aktualizáciách a hlavne o pridanej hodnote vytváraných riešení.

Ak však chceme vo výučbe pokryť čo najväčšiu časť uvedených tém, potrebujeme platformu, ktorá nám to umožní. V rámci príspevku predstavím platformu, ktorú sme vytvorili v prostredí Katedry počítačov a informatiky na Technickej univerzite v Košiciach. Platforma je postavená na otvorených hardvérových aj softvérových riešeniach, využíva kontajnerizáciu, a je horizontálne aj vertikálne škálovateľná. Vďaka návrhu nie je obmedzená na konkrétne technológie, ale je možné ju akokoľvek ďalej rozširovať či už o nové služby alebo o nové možnosti komunikácie. Je teda pripravená na kreatívne riešenia a študentské nasadenia.

PUBLIKOVANIE PRIESTOROVÝCH ÚDAJOV ÚGKK SR POD LICENCIOU OPEN DATA

DEKAN Tomáš (SK)

Úrad geodézie, kartografie a katastra Slovenskej republiky (ÚGKK SR) publikuje množstvo digitálnym priestorových údajov a webových mapových služieb, z ktorých mnohé sú poskytované ako otvorené údaje (Open Data) a sú tak voľne a bezodplatne dostupné širokej verejnosti.

Medzi najviac využívané patria produkty leteckého snímkovania a laserového skenovania (LLS), ktoré nachádzajú využitie v rôznych oblastiach. V roku 2017 začal ÚGKK SR v spolupráci s Národným lesníckym centrom letecké snímkovanie a tvorbu ortofotomozaiky SR. Jeden cyklus tvorby, počas ktorého sa nasnímkuje celé Slovensko, trvá tri roky a ročne sa spracuje 1/3 územia SR. Momentálne prebieha už 3. cyklus, kde raster ortofotomozaiky vo formáte TIFF má priestorové rozlíšenie 15 cm/pixel. Pri projekte LLS bol úspešne ukončený 1. cyklus, ktorý prebiehal v rokoch 2017 až 2023. V roku 2022 bol spustený 2. cyklus projektu LLS a údaje z prvých naskenovaných lokalít sú už dostupné na poskytovanie. Medzi poskytované produkty LLS patria klasifikované mračno bodov, digitálny model reliéfu (DMR) a digitálny model povrchu (DMP). DMR 5.0 a DMP 1.0 z 1. cyklu projektu LLS sú poskytované v rastrovom formáte TIFF s priestorovým rozlíšením 1 m. DMR 6.0 a DMP 2.0 z 2. cyklu projektu LLS sú poskytované v rastrovom formáte TIFF s priestorovým rozlíšením 0,5 m. Mračná bodov sú poskytované vo formátoch LAS a LAZ.

Z údajov ZBGIS (Základná báza údajov pre geografický informačný systém) sú voľne dostupné na stiahnutie administratívne hranice, geografické názvoslovie a ZBGIS Raster, ktorý reprezentuje nové štátne mapové dielo a predstavuje export údajov z databáz ZBGIS s priradenou kartografickou reprezentáciou v rastrovej forme vo formáte TIFF pre mierky 1:5 000 – 1:50 000.

Otvorené údaje katastra nehnuteľností sú poskytované prostredníctvom webovej aplikácie Atribúty katastrálneho operátu (`https://ako.vugk.sk`). Aplikácia poskytuje metaúdaje o súbore geodetických informácií katastra nehnuteľností a takisto umožňuje stiahnutie vektorových katastrálnych máp zo zvoleného katastrálneho územia vo formáte GeoPackage. Stiahnutý súbor obsahuje nasledujúce vrstvy s vybranými atribútmi: hranice katastrálneho územia, parcely C, parcely E a ZAPPRAR (hranice druhov pozemkov, ktoré nie sú hranicami parciel).

Ďalšou formou poskytovania priestorových údajov ZBGIS a katastra nehnuteľností sú webové mapové služby publikované podľa OGC štandardov WMS, WMTS a WFS.

ÚGKK SR takisto pripravuje a sprístupňuje údaje podľa smernice INSPIRE pre témy Administratívne hranice, Budovy, Dopravné siete, Geografické názvoslovie, Hydrografia, Katastrálne parcely, Ortometria a Výška. Okrem WMS služieb sú INSPIRE údaje dostupné na stiahnutie ako dátové balíčky vo formátoch GML, GeoPackage a ESRI GDB alebo v prípade rastrových údajov vo formáte TIFF.

ÚGKK SR poskytuje prístup k týmto údajom a mapovým službám prostredníctvom webovej stránky www.geoportal.sk a webovej aplikácie MAPKA – Mapový portál katastra (`https://zbgis.skgeodesy.sk/mapka/`).

APLIKACE PRO PREPROCESSING, IMPUTACI A ANALÝZU PROTEOMICKÝCH DAT

CHUPÁŇ Tomáš (CZ)

V dnešní době nám moderní technologie umožňují sbírat širokou škálu datových sad z různých oborů. Jedním z těchto oborů je proteomika, která v tomto tisíciletí zažívá působivý boom a prací s tímto typem dat se zabývá mnoho výzkumníků – avšak ne tolik, aby existovalo odpovídající množství nástrojů, které by jim každodenní úkoly usnadnily. Jde navíc o velmi specifickou, komplexní problematiku, u které se mnohdy ani lidé pracující přímo v tomto oboru neshodnou, jak při zpracování a analýze tohoto typu dat postupovat. Dalším úskalím je křivka učení, která při začátcích na tomto poli roste jen velmi pomalu a chvíli trvá, než se člověk zorientuje v tom, jaký je princip práce s těmito daty.

Jakýkoliv nástroj pro ulehčení práce s (nejen) proteomickými daty je tak jistě vítán – jednou z možností je Shiny aplikace vytvořená v prostředí R, které díky otevřenosti a umožnění přispět komukoliv svým balíčkem disponuje mj. také různými metodami využitelnými v proteomice. Cílem příspěvku je představit novou Shiny aplikaci proteoME, která slouží jako interaktivní nástroj pro preprocessing, imputaci a analýzu proteomických dat. Po importu datové sady a anotačních souborů nabízí základní vizualizaci dat, jejich transformaci (logaritmická a odmocninová) a normalizaci – data lze normalizovat pomocí mediánové či kvantilové normalizace, navíc je k dispozici metoda MBQN kombinující oba přístupy. Následujícím krokem je agregace na úroveň vzorků, při níž má uživatel k dispozici kromě několika možností postupu také vizualizace usnadňující volbu parametrů. Agregovaná data je také možné filtrovat dle procenta chybějících hodnot na několika úrovních, příp. chybějící hodnoty imputovat jednou z dostupných metod imputace (minimum vzorku, kNN, random forest). Samotná analýza takto připravených proteomických dat poskytuje několik možností testování shody distribuce abundancí proteinů mezi jednotlivými skupinami léčby – lze porovnávat libovolný počet takových skupin. Vždy je možné zvolit mezi parametrickým testem a jeho neparametrickou alternativou – při porovnávání dvou skupin jde o t-test a Wilcoxonův dvouvýběrový test, v případě více skupin má uživatel na výběr mezi ANOVA a Kruskal–Wallisovým testem. Výsledky analýzy ve formě detailní tabulky s širokou paletou možností exportu, řazení řádků i filtrace hodnot se snaží maximálně vyhovět potřebám uživatele, podobně jako hlavní grafický výstup analýzy – volcano plot, který si uživatel taktéž může přizpůsobit dle svých požadavků. V příspěvku bude názorně předvedeno také ukázkové zpracování syntetické datové sady v aplikaci proteoME s demonstrací všech výše uvedených kroků.

Programátorskou zajímavostí je modularita kódu této aplikace, kterou sice uživatel nevidí, ale pocítí ji vývojář v podobě velkého zjednodušení práce. Využity byly tzv. Shiny moduly, které si lze představit jako stavební kostky s konkrétním účelem, z nichž se výsledná aplikace sestaví. Další usnadnění přináší využití R balíčku golem, který obsahuje veškerou infrastrukturu pro každého, kdo by chtěl Shiny aplikaci vyvíjet přímo jako balíček v R – stejným způsobem byla vytvořena i aplikace proteoME.

Výsledná aplikace je dostupná na GitHubu (`https://github.com/TomChupan/proteoME`), odkud si ji každý uživatel R může nainstalovat jako jakýkoliv jiný R balíček. Také tam lze najít kompletní kód celé aplikace a mechanismy pro reportování problémů

či návrhy na zlepšení. Pro prezentační účely je proteoME dostupná také na serveru `shinyapps.io`, což umožňuje její spuštění a vyzkoušení všech funkcí i těm, kteří software R (zatím) nevyužívají.

MARIMO – REAKTÍVNY NOTEBOOK

KAUKIČ Michal (SK)

Marimo je nový prírastok v rodine webových nástrojov na spracovanie a znázorňovanie dát. Je napísané od piky, nezávisle od Jupytera. Snaží sa práve odstrániť niektoré nedostatky Jupyter notebookov, napr. chaotické vykonávanie buniek. Reaktivita znamená, že pri vykonaní bunky sa automaticky vykonajú bunky od nej závislé a aktualizujú sa ich výstupy. Grafické prvky (posuvníky, grafy, tabuľky. . .) sa používajú priamo, bez nutnosti písať funkcie pre ich vyvolávanie (callbacks). Marimo notebooky sú plne reprodukovateľné, poradie vykonávania buniek je deterministické. Ukážeme jednoduché príklady použitia (znázornenie dát pre jazdy newyorkských taxikárov).

ZNÁZORNENIE A INTERPRETÁCIA DÁT V MARIMO NOTEBOOKU (JAZDY TAXÍKOV V NEW YORKU)

KAUKIČ Michal (SK)

Aplikácia marimo pre znázornenie dát (ukázané v prednáške o marimo notebooku). Vytvoríme spolu grafy pre dáta o jazdách taxíkov v januári 2015 (nezávisle od marimo) a urobíme webovú interaktívnu aplikáciu, kde sa bude dať s tými grafmi pohrať. Jediné, čo budeme potrebovať, je webový prehliadač. Budeme využívať moduly polars, plotly, marimo. Programovanie v Pythone bude zrozumiteľné aj pre začiatočníkov. Hotovú aplikáciu si môžete pozrieť na `http://feelmath.eu:2025/` (dá sa pozrieť aj zdrojový kód).

ANALÝZY REGIONÁLNYCH ÚDAJOV O DEMOGRAFII A NEZAMESTNANOSTI KRAJÍN VYŠEHRADSKEJ SKUPINY V R A POSTGIS

PÁLENÍK Michal (SK)

Vykresľovanie máp je často vnímané ako technicky veľmi náročná úloha, čo v skutočnosti nie je. V príspevku si na konkrétnom datasete ukážeme, ako jednoducho vytvoriť graf či mapu so štatistickými údajmi. Použijeme výhradne slobodne licencovaný autorom vytvorený LAU1 dataset a softvér pod slobodnou licenciou (R, PostgreSQL a PostGIS).

Predstavíme LAU1 dataset dostupný na webstránke Inštitútu zamestnanosti `https://www.iz.sk/okresy s doi:10.5281/zenodo.6165135`. Obsahuje údaje o demografii (počty ľudí v jednotlivých vekových skupinách podľa rodu) a nezamestnanosti (počty uchádzačov o zamestnanie podľa rodu, veku, vzdelania, dĺžky evidencie) za 733 NUTS4 regiónov vyšehradskej skupiny, ktoré spolu majú 134 084 pozorovaní. LAU1 dataset tiež

obsahuje skratku okresu a hlavne polygón hraníc jednotlivých okresov, powiatov a járásov Slovenska, Česka, Poľska a Maďarska v súradnicovom systéme WGS84 spracovaný z databázy OpenStreetMap. Pre jednoduchosť používania je dataset je exportovaný do formátov ako csv, topojson, sql alebo shp.

Na spracovanie datasetu použijeme kombináciu open source programov PostgreSQL, PostGIS a R. Na uloženie údajov využijeme PostGIS pričom si predstavíme spôsoby načítania údajov do tejto relačnej databázy. Budeme vedieť využívať štandardné SQL príkazy, pričom pre ďalšie analýzy sú najdôležitejšie agregátne funkcie (st_union, group by), prepájanie tabuliek (join) a vyberanie podmnožín (where). Relačná databáza umožňuje rýchle, flexibilné a užívateľský prítulné uloženie údajov.

Samotné vykresľovanie máp budeme realizovať v štatistickom balíku R. Predstavíme si základné príkazy Rka, spôsoby prepojenia PostGISu a R na účely stiahnutia predspracovaných údajov, základné štatistické analýzy a najmä vykresľovanie grafov a máp. Použijeme štandardné knižnice, vďaka čomu je vykreslenie mapy iba niekoľkoriadkový Rkový kód. Záver prednášky je vytvorenie mapy miery nezamestnanosti Žilinského kraja a Česko-Slovenska.

OTEVŘENÁ GEODATA JAKO KLÍČ PRO INFORMOVANOU SPOLEČNOST V ČESKU

PLÁNKA Michal, BURIAN Jaroslav (CZ)

Data se stávají „ropou 21. století“ a jsou již dnes jedním z klíčových prvků pro správné a přesnější rozhodování o správě a rozvoji Česka. Zejména geografická data (geodata) mají nejen v Česku nezastupitelnou podstatu. Geodata, jsou v tomto ohledu mimořádně důležitá, protože poskytují klíčové informace o veřejných službách, infrastruktuře, životním prostředí a mnoha dalších oblastech. Jedná se především o data z nejrůznějších státních registrů a databází, jejichž využití je veřejnosti k dispozici bezplatně. Mnoho těchto datových zdrojů je dnes dostupné také formou otevřených dat (open dat), která jsou základem pro transparentní a participativní správu mezi státními orgány a veřejností. Díky otevřeným datům mohou občané, instituce nebo firmy snadno získávat informace a monitorovat činnost veřejné správy nebo využívat data pro vzdělávací či komerční účely. Centrálním bodem otevřených dat je Národní katalog otevřených dat (`https://data.gov.cz`), který shromažďuje otevřená data poskytovaná různými institucemi a organizacemi. Zpracování těchto dat však již vyžaduje jistou technickou znalost, což může být pro některé potenciální uživatele limitující.

Geodata, definovaná jako data s implicitním nebo explicitním vztahem k místu na Zemi, zahrnují široké spektrum informací o geografické poloze a charakteristikách přírodních a antropogenních jevů. V České republice jsou tato data shromažďována a poskytována různými institucemi, jako je Český statistický úřad (ČSÚ) a Český úřad zeměměřický a katastrální (ČÚZK). Otevřená data umožňují jejich volné a opakované využití bez finančních a technických překážek, což zvyšuje jejich hodnotu a podporuje transparentní a participativní správu.

Nejčastěji využívanými daty jsou především statistická data, avšak stále větší význam získávají data z čidel a senzorů (například měření hluku a znečištění), stejně jako data

získaná z dronů, letadel nebo družic (například termální snímky) a další různé typy prostorových dat. Kromě samotných dat je nezbytné zaměřit se také na jejich analýzu, vizualizaci (například prostřednictvím map, mapových aplikací nebo dashboardů) a správnou interpretaci. V tomto ohledu jsou neocenitelnými pomocníky geografické informační systémy (GIS), které nabízejí širokou škálu možností pro správu, analýzu a vizualizaci všech dostupných typů dat. Některé státní instituce v České republice provozují webové mapové portály, které zastupují jejich odbornou činnost a zpřístupňují specifická tematická data široké veřejnosti.

Český statistický úřad (ČSÚ) je klíčovým poskytovatelem socioekonomických statistických dat, která jsou snadno dostupná prostřednictvím Veřejné databáze (`https://vdb.czso.cz`). Tento portál umožňuje stahování dat za různé územní celky a nabízí jednoduchou vizualizaci dat ve formě grafů a map. Další významný nástroj ČSÚ, Statistický geoportál (`https://geodata.statistika.cz`), zpřístupňuje data z posledního cenzu z roku 2021, včetně informací o dojížďce a vybraných statistických dat za obce.

Český úřad zeměměřický a katastrální (ČÚZK) poskytuje klíčová prostorová data, včetně základních map, leteckých snímků, katastrálních map a výškopisných dat. Tato data jsou nově dostupná jako otevřená data a lze je prohlížet v Geoprohlížeči (`https://ags.cuzk.cz/geoprohlizec`). Zajímavou aplikací od ČÚZK jsou Analýzy výškopisu (`https://ags.cuzk.cz/av`), které zobrazují podrobné 3D digitální modely získané přesným leteckým laserovým snímkováním celého Česka. Aplikace umožňuje analyzovat vybrané vlastnosti terénu, jako je sklon a orientace, nebo hodnotit viditelnost z vybraných bodů.

Kromě těchto hlavních poskytovatelů otevřených dat existuje celá řada dalších státních institucí, které provozují vlastní geoportály se specifickými tematickými daty a interaktivními mapami. Například portál Otevřená data AOPK ČR (`https://gis-aopkcr.opendata.arcgis.com`) nabízí data o ochraně přírody, Geoportál Národního památkového ústavu (`https://geoportal.npu.cz`) poskytuje informace o kulturním dědictví, a Atlas veřejné správy (`https:// portal-vnitro.hub.arcgis.com`) zahrnuje různé veřejné služby.

Za zmínku rovněž stojí interaktivní přehledový informační systém Mapy kriminality (dále jen „mapa kriminalita“), který je provozován Policií České republiky a zobrazuje vybranou přestupkovou a trestnou činnost evidovanou Policií České republiky. Mapa kriminality je určena veřejnosti a má přispět k prevenci kriminality. Zobrazuje delikty (přestupky a vybrané trestné činy), kterými se v daném období Policie České republiky zabývala, nad mapovým podkladem, nezobrazuje přesné místo spáchání, pouze podbarvenou oblast, v níž k protiprávnímu jednání došlo.

Cílem příspěvku je ukázat obrovský potenciál využití geografických dat v Česku pro zlepšení kvality života, informovanosti a vzdělání obyvatel, efektivní využití zdrojů a podporu udržitelného rozvoje. Integrace otevřených dat, prostorových dat, analýzy dat a geografických informačních systémů umožňuje lidem lépe porozumět svému okolí a efektivněji plánovat a řídit své činnosti. Dosavadní přístupy založené na intuici, zkušenostech a expertních odhadech mohou být doplněny o objektivní datový pohled, který zefektivňuje způsob fungování a rozhodování v mnoha subjektech.

ZÁSUVNÝ MODUL DYNA CROP

RŮŽIČKA Jan, KALÁB Oto, RŮŽIČKOVÁ Kateřina (CZ)

S rozvojem družicových systémů je k dispozici stále větší množství dat, která nám mohou pomoci s analýzou stavu krajiny. S využitím těchto dat, která se nejčastěji dělí do čtyř kategorií: multispektrální, hyperspektrální, radarová a lidarová, je možné vytvářet různé tematické výstupy. Příkladem může být jednoduchý normalizovaný vegetační index, známý pod zkratkou NDVI, který kombinací červeného a blízkého infračerveného pásma dokáže uživateli zobrazit indikaci zdravotního stavu vegetace. Zásadní výhodou družicových dat je také to, že je možné získávat záznamy v pravidelných intervalech a sledovat tak vývoj území. Řada dat je k dispozici i zcela zdarma, příkladem může být systém Sentinel, provozovaný agenturou ESA (European Space Agency). Získání a analýza dat však může být pro nezkušeného uživatele opravdovým problémem. Z tohoto důvodu vznikají různé aktivity, které se snaží práci s těmito zdroji zjednodušit.

Zásuvný modul Dyna Crop slouží pro komunikaci s API společnosti World From Space. Toto API umožňuje podrobnou analýzu družicových dat pro účely zemědělství. Zásuvný modul byl vyvinut v rámci skupiny GISMentors ve spolupráci se společností World From Space. Umožňuje využít QGIS k vytvoření zájmových polygonů a k vizualizaci výstupních indexů družicových dat. API nabízí výpočet celé řady vegetačních indexů a jiných produktů a také sestavení zonace území dle různých kritérií.

Úplný výčet produktů je k dispozici v dokumentaci na `https://docs.dynacrop.space/docs/#/products`. Zde uvedeme pouze jejich seznam (ARI1 – Anthocyanin Reflectance Index 1, CAR_RE – Red-edge Carotenoid Reflectance Index, CCC – Canopy Chlorophyll Content, CCI – The Chlorophyll Carotenoid Index, CL_RE – Red-Edge Chlorophyll Index, CWC – Canopy Water Content, EVI – Enhanced Vegetation Index, FAPAR – Fraction of Photosynthetically Active Radiation, IRECI – Inverted Red-Edge Chlorophyll Index, LAI – Leaf Area Index, MNDWI – Modified Normalized Difference Water Index, MSAVI2 – Modified Soil-adjusted Vegetation Index, MTVI2 – Modified Triangular Vegetation Index – Improved, NDMI – Normalized Difference Moisture Index, NDRE – Normalized Difference Red Edge Index, NDREX – Normalized Difference Red Edge Index, NDVI – Normalized Difference Vegetation Index, NDWI – Normalized Difference Water Index, NMDI – Normalized Multi-band Drought Index, PRI – Photochemical Reflectance Index, PVI – Perpendicular Vegetation Index, SI – Salinity Index, SIPI – Structure Insensitive Pigment Index, SMI – Soil Moisture Index, SOC_VIS – Soil Organic Carbon, SOC_SWIR – Soil Organic Carbon, WIW – Water In Wetlands, YRSI – Yellow Rust Spore Index) a stručně popíšeme některé z nich.

ARI1 – Anthocyanin Reflectance Index 1 umožňuje detekci, jak rostliny reagují na zátěž z prostředí v podobě sucha, nedostatku výživy a vlivu patogenů. Jako zdroj je využíván systém družic Planet a jimi pořízená multispektrální data.

CL_RE – Red-Edge Chlorophyll Index umožňuje detekovat stav chlorofylu ve strukturách rostliny a lépe než index NDVI poukázat na vývoj rostliny, zejména v rané fázi růstu. Jako zdroj je využíván systém družic Sentinel a jimi pořízená multispektrální data.

NMDI – Normalized Multi-band Drought Index umožňuje detekci přítomnosti vodní složky v listech a půdě. Jako zdroj je využíván systém družic Sentinel a jimi pořízená multispektrální data.

SMI – Soil Moisture Index umožňuje detekci vlhkosti půdy. Jako zdroj je využíván systém družic Sentinel a jimi pořízená radarová data (SAR).

Zásuvný modul pak tyto výsledky vizualizuje v podobě rastrových dat a grafů, které ukazují vývoj území v čase. V rámci prezentace bude představeno API, zásuvný modul a také se podíváme na fungování samotného modulu. Pro posluchače může být například zajímavé, jakým způsobem je řešeno asynchronní zpracování realizovaných požadavků na API.

OTVORENÝ SOFTVÉR PRE ROZVÍJANIE DIGITÁLNYCH ZRUČNOSTÍ 21. STOROČIA

ŠECHNÝ Martin (SK)

Aké spôsobilosti sú potrebné pre štúdium, prácu a život v 21. storočí? Chápeme tým najmä komunikáciu, spoluprácu, tvorivosť a kritické myslenie. Tieto a ďalšie spôsobilosti môžeme priradiť k viacerým gramotnostiam. Predmetom digitálnej gramotnosti je najmä rozvíjanie základných digitálnych zručností využiteľných v digitálnej spoločnosti vo všetkých oblastiach. Špeciálne v informatike sú digitálne zručnosti nevyhnutné pri formovaní výpočtového myslenia. DigComp 2.2, rámec pre digitálnu gramotnosť, má päť problémových oblastí: dáta a informácie, komunikácia a spolupráca, vytváranie digitálneho obsahu, bezpečnosť, riešenie problémov (VOURIKARI, R., KLUZER, S., PUNIE, Y., 2022, DigComp 2.2: The Digital Competence Framework for Citizens. Luxembourg: Publications Office the European Union. ISBN 978-92-76-48882-8, EUR 31006 EN, JRC128415, doi: 10.2760/115376, `https://publications.jrc.ec.europa.eu/repository/bitstream/JRC128415/JRC128415_01.pdf`, `https://joint-research-centre.ec.europa.eu/digcomp_en`). Digitálne zručnosti dostávajú svoj zmysel pri použití digitálnych nástrojov (IT) a digitálnych dát. Otvorené vzdelávanie všeobecne má využívať otvorené vzdelávacie zdroje, prednostne otvorené IT nástroje, teda otvorený softvér a otvorený hardvér, spolu s otvorenými dátami.

Prvá problémová oblasť DigComp, dáta a informácie, zahŕňa vyhľadanie, usporiadanie, uloženie dát, informácií a digitálneho obsahu, overenie dôveryhodnosti a spoľahlivosti zdroja dát. Najčastejšie používanými nástrojmi sú webový prehliadač, vyhľadávač informácií a rôzne katalógy informácií. Dáta ukladáme prevažne vo forme súborov a na prácu s nimi potrebujeme operačný systém, balík kancelárskych aplikácií, prehliadač multimédií, informačný alebo databázový systém. Každý používateľ so základnými digitálnymi zručnosťami má rozlišovať otvorený a uzavretý softvér, vymenovať niekoľko alternatív a vybrať si konkrétny digitálny nástroj, vhodný pre danú situáciu. Druhá problémová oblasť, komunikácia a spolupráca, pokrýva digitálne služby a médiá. Každým rokom pribúdajú nové a sofistikovanejšie nástroje pre komunikáciu a spoluprácu, použitie digitálnej identity v digitálnom prostredí. Stretávame sa prevažne s komerčnými digitálnymi službami, pretože primárnym cieľom podnikateľa na digitálnom trhu je dosahovať zisk. Avšak komerčná digitálna služba si nevyžaduje nutne komerčný digitálny nástroj. Podnikať sa dá aj s otvoreným softvérom. Tretia problémová oblasť, vytváranie digitálneho obsahu, popisuje vytváranie, editovanie a spájanie digitálneho obsahu viacerých formátov pomocou vhodného digitálneho nástroja, pri dodržiavaní autorských práv a etiky. Do tejto

problémovej oblasti patrí aj vytváranie programu, automatizácia úloh, výpočet pomocou počítača. Možnosť vytvoriť otvorený zdrojový kód je prejavom slobody myslenia.

Bezpečnosť je problémová oblasť, ktorej dôležitosť stále rastie, pretože náš život sa stále viac stáva digitálnym, produkujeme väčšie množstvo digitálnych dát, využívame viac digitálnych služieb zasahujúcich do súkromia a spoliehame sa na automatizáciu. Aké sú výhody a aké sú nevýhody otvoreného softvéru pri posudzovaní bezpečnosti? Môžeme dôverovať vyhláseniu o bezpečnosti a súkromí pri komerčnej digitálnej službe, ktorá dosahuje zisk hlavne z predaja osobných údajov používateľa? Posledná problémová oblasť, riešenie problémov, má dva pohľady – riešenie bežných problémov s digitálnou technikou a stratégie riešenia rôznych problémov pomocou digitálnych nástrojov. Otvorený softvér má výhodu – právo použiť softvér na akýkoľvek účel, prístupnosť otvoreného zdrojového kódu, právo editovať zdrojový kód, právo šíriť pôvodný alebo modifikovaný softvér. Obdobne to platí pre otvorený hardvér, otvorené dáta a otvorené vzdelávacie zdroje.

Aktuálnym technologickým trendom je využitie strojového učenia a umelej inteligencie v digitálnych nástrojoch vo všetkých problémových oblastiach. Do základnej digitálnej gramotnosti patrí efektívne, bezpečné, etické a zodpovedné použitie takého nástroja, kritické posúdenie dôveryhodnosti vytvoreného digitálneho obsahu.

IT špecialisti sa líšia od ostatných tým, že majú pokročilé digitálne zručnosti. Perspektívne špecializácie dnes sú napríklad dátový analytik, manažér digitálnej transformácie, špecialista informačnej a kybernetickej bezpečnosti, špecialista na kvantové technológie. Vo všetkých špecializáciách má svoj význam otvorený softvér. Vzdelávací program a certifikácia LPI Open Source Essentials poskytuje základné princípy pre používanie aj vytváranie otvoreného softvéru – princípy softvérového inžinierstva, typy licencií, biznis model, projektový manažment, komunikácia a spolupráca (LPI Open Source Esentials, `https://www.lpi.org/articles/lpi-launches-the-open-source-essentials-education-program/`, `https://wiki.lpi.org/wiki/Open_Source_Essentials_Objectives_V1.0`, `https://www.lpi.org/our-certifications/open-source-essentials/`).

BDTI Essentials je otvorený vzdelávací program pre špecializáciu dátový analytik s použitím otvoreného softvéru. Kurz vysvetľuje spôsoby získavania dát, preskúmania, čistenia, transformácie, spájania a ukladania dát, princípy dátovej analytiky, vizualizácie dát. Používajú sú viaceré nástroje, programovacie jazyky a algoritmy. Populárne sú napr. Python, R, KNIME. BDTI poskytuje vzdialené virtualizované výpočtové prostredie s podporou všetkých menovaných nástrojov a jazykov (Big Data Test Infrastructure – BDTI, `https://big-data-test-infrastructure.ec.europa.eu/index_en`, `https://big-data-test-infrastructure.ec.europa.eu/resources/courses-and-training/bdti-essentials-course_en`.

VÝCHODISKÁ PRE VYUŽITIE AI VO VZDELÁVANÍ

ŠECHNÝ Martin (SK)

Umelá inteligencia dnes ešte nie je dostatočne inteligentná. Ako ju definovať? Je to schopnosť počítača napodobňovať rozumové schopnosti a správanie sa človeka.

Generatívna umelá inteligencia využívajúca jazykový model vytvára odpoveď skladaním slov tak, aby s najväčšou pravdepodobnosťou jazykovo pasovala k otázke. Taká

odpoveď nemusí byť pravdivá a logická, lebo nie je myšlienkou. Dnešný stroj s jazykovým modelom nemyslí. Strojové učenie je však dobré pre jazykový preklad.

Generatívna umelá inteligencia môže vytvoriť obrázok, video. Generatívna znamená vytvárajúca. Je aj tvorivá? Je fyzika tvorivá len preto, že z vody v zime vytvára pekné unikátne snehové vločky vo veľkom množstve?

Príchod generatívnej umelej inteligencie je zmena paradigmy v technológii, automatizácii, etike, sociológii a iných oblastiach. Musíme zmeniť naše očakávania a hodnoty. Povolania budú iné, ako dnes. Kvôli AI niektoré pracovné pozície zaniknú a vzniknú iné, je to ďalšia priemyselná revolúcia.

Vo vzdelávaní je nutná zmena – musíme pripraviť žiaka/študenta na život v novej digitálnej dobe, musí si byť vedomý dobrých aj zlých vlastností AI nástroja a má vedieť ho použiť správne, efektívne, bezpečne, eticky a zodpovedne. Nesprávne použitie môže viesť ku generovaniu nezmyslov alebo nebezpečného obsahu. Človek bude pri používaní umelej inteligencie efektívnejší a výkonnejší, ale na druhej strane stráca samostatnosť a naučené schopnosti. Človek je lenivý. Dostupnosť generatívnej umelej inteligencie znižuje nevyhnutnosť používať vlastný rozum a vlastnú tvorivosť.

Digitálna gramotnosť je vybraná sada preukázaných schopností jednotlivca sebaisto, kriticky a zodpovedne využívať digitálne technológie pre život, učenie sa a prácu v digitálnej spoločnosti. Digitálna gramotnosť zahŕňa použitie umelej inteligencie prierezovo vo všetkých vzdelávacích oblastiach. Príklady z obsahu vzdelávania pre ZŠ v novom štátnom vzdelávacom programe:

Vyhľadať, usporiadať a uložiť dáta, informácie a digitálny obsah. Porozumenie, že umelá inteligencia ako taká nie je ani dobrá, ani zlá a jej dopad záleží od toho ako a kým je navrhnutá a používaná. Porozumenie, že mnohé digitálne prostredia využívajú mechanizmy ako ovplyvňovanie, gamifikácia a manipulácia, s cieľom ovplyvniť správanie používateľov. Rozpoznanie vplyvu umelej inteligencie na výsledky vyhľadávania a informácie v sociálnych sieťach (sociálne bubliny, emotívne podfarbené informácie). Rozlíšenie nepresných informácií, nepravdivých informácií, digitálneho obsahu vytvoreného umelou inteligenciou (deep fake) a dezinformácií (s úmyslom oklamať). Porozumenie, že dáta používateľa môžu byť použité na trénovanie umelej inteligencie. Identifikovanie niektorých príkladov použitia umelej inteligencie pri rozpoznaní hlasu a obrazu. Použitie strojového prekladu pri cudzojazyčnom návode na riešenie problému. Použitie digitálnych technológií pre inovácie a vytváranie nových hodnôt (umelá inteligencia, internet vecí, robotika).

Informatika je študijný odbor, veda a priemyselné odvetvie – zaoberajú sa automatizovaným spracovaním digitálnych dát. Informatika ako vyučovací predmet má vysvetľovať, ako digitálna technológia funguje. Umelá inteligencia je postavená na niekoľkých princípoch (rozpoznanie vzoru, strojové učenie, štatistické rozhodovanie), na niekoľkých technológiách (umelá neurónová sieť, genetický alebo evolučný algoritmus) a na dostatočnom množstve dát.

Ako hodnotiť žiaka/študenta za riešenie úlohy, ktoré vytvoril pomocou AI nástroja? Ako určiť autorstvo generovaného dokumentu? Nemá zmysel hodnotiť iba odovzdaný dokument. Dôležitejšia je argumentácia, princípy a vysvetlenie riešenia. Úspešná obhajoba riešenia zahŕňa tiež uplatnenie mäkkých zručností.

Aké AI nástroje použiť vo vzdelávaní? Niektoré nástroje sú dostupné slobodne, niektoré bezplatne s registráciou za poskytnutie osobných údajov, niektoré sú platené, niektoré integrované do väčšej služby. Potrebujeme vybrať vhodné nástroje pre daný účel, preštudovať obchodné podmienky. Treba zabezpečiť platformovú neutralitu a vyhnúť sa uviaznutiu u monopolného dodávateľa (vendor lock-in). Preferovaný by mal byť otvorený softvér a otvorené dátové modely. Populárne digitálne nástroje získavajú veľké množstvo osobných údajov používateľa, čo je v rozpore s požiadavkami na ochranu súkromia a požiadavkami na zabezpečenie výchovy (najmä detí do 15 a do 18 rokov).

Digitálna transformácia vzdelávania je zmena výchovno-vzdelávacieho procesu a procesov administratívy školy z papierových na procesy riadené dátami a algoritmami. Cieľový stav: inteligentná otvorená škola. AI môže byť užitočnou technológiou pre personalizovaný/adaptívny výchovno-vzdelávací proces a efektívne riadenie školy. AI nástroje môžu učiteľovi pomôcť vytvárať obsah vzdelávania, testovanie a hodnotenie. Motivácia žiaka/študenta zrejme aj naďalej ostatne hlavnou úlohou učiteľa-človeka.

Zdroje a odkazy.

Neuroscientist explores how ChatGPT mirrors its users to appear intelligent techxplore.com/news/2023-03-neuroscientist-explores-chatgpt-mirrors-users.html.
Large language models are biased. Can logic help save them? https://techxplore.com/news/2023-03-large-language-biased-logic.html.
Generative AI like ChatGPT reveal deep-seated systemic issues beyond the tech industry https://techxplore.com/news/2023-03-generative-ai-chatgpt-reveal-deep-seated.html.
AI could take your job, but it can also help you score a new one with these simple tips https://techxplore.com/news/2023-03-ai-job-score-simple.html.
Program Digitálna Európa (2021 – 2027), https://digital-strategy.ec.europa.eu/en/activities/digital-programme.
Nariadenie o digitálnych službách (2022), https://commission.europa.eu/strategy-and-policy/priorities-2019-2024/europe-fit-digital-age/digital-services-act-ensuring-safe-and-accountable-online-environment_en, https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32022 R2065&qid=1666857835014.
Biela kniha o umelej inteligencii (2020), https://commission.europa.eu/document/d2ec4039-c5be-423a-81ef-b9e44e79825b_sk.
Akčný plán digitálneho vzdelávania (2021 – 2027), https://education.ec.europa.eu/focus-topics/digital-education/action-plan.
DigComp 2.2, https://joint-research-centre.ec.europa.eu/digcomp_en,
https://publications.jrc.ec.europa.eu/repository/handle/JRC128415,
https://publications.jrc.ec.europa.eu/repository/bitstream/JRC128415/JRC128415_01.pdf.
Stratégia digitálnej transformácie Slovenska 2030, https://www.mirri.gov.sk/wp-content/uploads/2019/06/Strategia-digitalnej-transformacie-Slovenska-2030.pdf
Akčný plán digitálnej transformácie Slovenska na roky 2023 – 2026, https://www.mirri.gov.sk/wp-content/uploads/2023/01/APDTS-2023-2026.pdf.

INTEGRÁCIA NÍZKONÁKLADOVÝCH A OTVORENÝCH TECHNOLÓGII VO VÝVOJI GEODETICKÉHO VYBAVENIA

ŠPERKA Jakub (SK)

Práca spracováva problematiku implementácie otvorených a nízkonákladových IoT riešení v geodézii. Skúma možnosti zapojenia internetu vecí do tvorby funkčných prototypov vybavenia použiteľného na geodetické činnosti, jeho interakciu a možnosti ovládania. Práca popisuje jednotlivé aspekty tvorby funkčného prototypu elektronického príslušenstva pre geodetické odrazné hranoly, ktorá umožní automatické natáčanie cieľa v rámci geodetickej siete, pri realizácii merania s univerzálnou meracou stanicou. Produktom vývoja, ktorý práca opisuje, je funkčný prototyp spolu z príslušným technickým a programovým riešením založeným na platforme ESP32. Vývoj programu ovládacieho panelu v jazyku Python zahŕňa prehľadné používateľské, pre intuitívne ovládanie a modifikáciu. Produkt je podrobený praktickému testovaniu vo vybranej aplikačnej oblasti geodézie a geoinformatiky. Počas celého vývoja jednotlivých súčasti systému je braný ohľad na otvorenosť, jednoduchosť a modifikovateľnosť riešenia, aby mohli byť implementované pripomienky užívateľov a zachovaná cenová dostupnosť v porovnaní s komerčnými riešeniami. Vývoj príslušného produktu s ohľadom na praktickosť použitia v teréne a ľahkú dostupnosť, podporuje jeho využitie v praxi, a tým môže napomôcť zefektívniť vykonávanie niektorých geodetických činností v špecifických podmienkach.

DISKUSIA O OPEN SOURCE SOFTVÉRI V KONTEXTE OTVORENEJ VEDY NA SLOVENSKU

STOŽICKÁ Zuzana (SK)

Softvér s otvoreným zdrojovým kódom zohral významnú úlohu pri vzniku hnutia otvorenej vedy a stále k napĺňaniu jej cieľov prispieva tvorbou dostupných IT nástrojov, ktoré majú potenciál ušetriť prostriedky jednotlivcom aj inštitúciám financovaným z verejných zdrojov. Medzi kľúčové príklady v tejto oblasti patrí softvér umožňujúci prevádzku inštitucionálnych repozitárov, systémy na podporu vedeckého publikovania v režime otvoreného prístupu Open Journal System a Open Monograph Press, alebo štatistický balík R. Zdieľanie dát a zdrojového kódu ako doplňujúcich materiálov k publikáciám tiež prispieva ku zvýšeniu transparentnosti a reprodukovateľnosti výskumu. Softvér s otvoreným zdrojovým kódom má podporu Európskej komisie (Open Source Software Strategy).

Napriek tomu v slovenskej akademickej komunite mimo vedných odborov, ktoré sú s informačnými technológiami tradične úzko späté, panuje v súvislosti s open source softvérom nízke povedomie (občas v kombinácii s podceňovaním alebo obavami, niekedy aj u vedeckých pracovníkov, ktorí open source softvér bežne používajú a nevedia, že ich pracovný nástroj spadá do tejto kategórie), chýba systematická podpora a cielené vzdelávanie o tejto téme.

K úlohám Centra vedecko-technických informácií SR v rámci Národnej stratégie pre otvorenú vedu 2021-2028 patrí popri podpore open access publikovania, otvorených vzdelávacích zdrojov, rozvoja občianskej vedy či zdieľania a využívania otvorených dát a doplňujúcich materiálov k výskumu, aj podpora využívania otvorených IT nástrojov (Strategická oblasť 6).

Tento príspevok si kladie za cieľ predstaviť aktivity CVTI SR v oblasti otvorenej vedy (kurzy, webináre, otvorené vzdelávacie zdroje ako Sprievodca svetom vedeckého publikovania, príručky k licenciám Creative Commons a príručky pre doktorandov Cesta k otvorenej vede a aktuálne novo preložená Otvorená veda – Zdrojový kód a softvér, ktoré sú bezplatne prístupné online pre všetkých záujemcov) a otvoriť diskusiu s komunitou podporovateľov, tvorcov a používateľov open source softvéru o pripravovanom mapovaní otvorených IT nástrojov používaných v slovenskom akademickom prostredí. Chceli by sme zhromaždiť a prehľadne sprístupniť príklady dobrej praxe, informácie a otvorené vzdelávacie zdroje súvisiace s open source softvérom, ideálne v slovenskom jazyku, aby sme zviditeľnili softvér s otvoreným zdrojovým kódom aj v tých oblastiach slovenskej vedy, kde zatiaľ nie je rozšírený, poskytli slovenským vedcom ucelenú ponuku, možnosť učiť sa (alebo použiť tieto materiály pri vyučovaní študentov) a tiež súbor argumentov, ktorý manažérom vedy a tvorcom politík dokáže opodstatnenie systematickej podpory open source softvéru v rámci ich inštitúcií a oblastí pôsobnosti.

OPEN SOURCE PRE PODPORU INSPIRE NA SLOVENSKU

TUCHYŇA Martin, KOŠKA Martin, MOZOLÍK Peter (SK)

Intro. Budovanie a prevádzka národnej infraštruktúry priestorových informácií (NIPI) (`https://inspire.gov.sk`) ako Slovenského príspevku k Európskej iniciatíve INSPIRE (`https://inspire.ec.europa.eu`) sú v úzkom v spojení s využívaním otvoreného softvéru už viac ako 15 rokov. Hlavným poslaním INSPIRE je posilniť zdieľanie a využívanie priestorových informácií, ktoré majú dopad na životné prostredie. V rámci tohto poslania zohráva softvér s otvoreným kódom dôležitú úlohu.

Životný cyklus. V rámci úsilia zabezpečenia harmonizovaného zdieľania priestorových informácií medzi ich poskytovateľmi a používateľmi je potrebné zohľadniť všetky fázy životného cyklu údajov, od ich zberu/tvorby, cez následné spracovanie, harmonizáciu, harmonizované zdieľanie (publikáciu) až po samotné využitie a následnú archiváciu. V rámci implementácie INSPIRE na Slovensku boli postupne nasadzované a vyvíjané rôzne softvérové riešenia – od proprietárnych, cez open source až po špecifické „custom“ aplikácie. V súčasnosti je možné skonštatovať, že z pohľadu koordinácie INSPIRE na Slovensku je väčšina bežiacich a vyvíjaných riešení postavená nad open source, prípadne vlastným (custom) vývojom s použitím open source komponentov s perspektívou jeho sprístupnenia pod otvorenou licenciou s cieľom zlepšiť tieto softvérové riešenia a podporiť ich opätovné využitie.

Infraštruktúra. Základom pre rozvoj a prevádzku softvérových riešení je serverová infraštruktúra, ktorá dnes vytvára nevyhnutný základ pre webovo orientované GIS aplikačné riešenia. Pre podporu implementácie INSPIRE na Slovensku bola v prostredí Vládneho cloudu vytvorená infraštruktúra využívajúca prostredia open source operačných systémov (Ubuntu, CentOS), aplikačné služby (napr. Nextcloud, INSPIRE validator), ktoré

vytvárajú ucelenú GeoICT platformu GeoCloud (`https://geocloud.gov.sk`). Pre atomizáciu konfigurácie a prevádzku infraštruktúry sa využíva Ansbile. Okrem týchto sú v infraštruktúre nasadené ďalšie overené open source softvérové riešenia (NGINX, Haproxy, BIND, OpenSMTP, Borgbackup, Fluentd, Squid a ďalšie).

Správa údajov, transformácie, publikácia služieb a dokumentácia metaúdajmi. Údaje predstavujú kľúčovú časť infraštruktúry a pre ich správu platforma GeoCloud využíva viaceré otvorené softvérové komponenty. Pre správu priestorových údajov sú využívané technológie PostgreSQL/PostgGIS. Transformáciu a publikáciu v spolupráci s PostGIS zabezpečuje Geoserver s app schema pluginom, pričom pre podporu rýchlejšieho vykresľovanie využitím techniky dlaždicovania je využívané riešenie Mapproxy. Metaúdaje slúžia na popis a uľahčenie vyhľadávania údajov, služieb a aplikácií. Zároveň metaúdaje predstavujú dôležitý spojovací prvok medzi poskytovateľmi a používateľmi digitálneho priestorového obsahu.

Monitoring. Pre zabezpečenie prehľadu o stave infraštruktúry zohráva kľúčovú úlohu systémový a prevádzkový monitoring. Tu vstupujú riešenia od Zabbix (monitoring infraštruktúry a platformy), Elasticsearch, Heartbeat & Kibana (monitoring dostupnosti INSPIRE služieb).

Využívanie. Platforma Geocloud takisto vytvára prostredie pre prevádzku a rozvoj 3 hlavných INSPIRE aplikácií:

– Register priestorových informácií (`https://rpi.gov.sk`) – (frontend Angular, backend FastAPI + PostgreSQL/PostGIS) — určený prioritne ich poskytovateľom.

– Národný Geoportál (`https://geoportal.gov.sk`) – (frontend Vue.js / Vuetify, backend FastAPI + PostgreSQL/PostGIS) – zameraný na používateľov.

– Websídlo NIPI (https://inspire.gov.sk) – (frontend Next) – poskytujúce informácie o problematike INSPIRE na Slovensku.

– CMS systém (PHP + PostgreSQL).

Prínosy. Využívanie Open source technológií z pohľadu MŽP SR prinieslo za doterajšie obdobie nemalé pozitívne prínosy, od úspory verejných zdrojov, silnej nezávislosti a obmedzenia vendor locku, cez vytvorenie možností spolupráce so špecifickými externými expertmi bez potreby zložitých postupov štúdia prostredia a špecifických požiadaviek konkrétnych softvérových nástrojov.

Výzvy. Samozrejme využívanie Open source technológií prináša i výzvy. Najvýraznejšou je obmedzená expertízna ponuka v segmente ľudských zdrojov, niektoré obmedzenia v oblasti užívateľského komfortu predovšetkým pri frontendových rozhraniach, prípadne obmedzenej komunitnej podpory niektorých Open source produktov. Napriek tomu je možné skonštatovať, že sa zatiaľ orientácia na podporu softvérových riešení nad otvoreným kódom ukazuje v podmienkach MŽP SR pre oblasť podpory INSPIRE ako dobré rozhodnutie, ktoré hlavne s odstupom času začína prinášať svoje prínosy.